FINITE ELEMENTS IN PLASTICITY:

Theory and Practice

D. R. J. OWEN
E. HINTON
Department of Civil Engineering
University College of Swansea, U.K.

Pineridge Press Limited
Swansea, U.K.

FINITE ELEMENTS IN PLASTICITY:

Theory and Practice

D. R. J. OWEN
E. HINTON
Department of Civil Engineering
University College of Swansea, U.K.

Pineridge Press Limited
Swansea, U.K.

First published 1980 by

Pineridge Press Limited
91 West Cross Lane, West Cross, Swansea U.K.

ISBN 0-906674-05-2

OWEN, D. R. J.
Finite Elements in Plasticity
1. Elasticity
2. Plasticity
3. Finite Element Method
I. Title II. HINTON, E.
620.1'123 TA418
ISBN 0-906674-05-2

Printed and bound in Great Britain by
REDWOOD BURN LIMITED
Trowbridge and Esher

Contents

Preface

The purpose of this text is to present and demonstrate the use of finite element based methods for the solution of problems involving plasticity. As well as the conventional quasi-static incremental theory of plasticity, attention is given to the slow transient phenomenon of elasto-viscoplastic behaviour and also to dynamic transient problems. We make no pretence that the text provides a complete treatment of any of these topics but rather we see it as an attempt to present numerical solution techniques, which have been well tried and tested, for selected important areas of application.

In our earlier books on finite elements we have concentrated on linear applications. Here we attempt the much more daunting task of introducing, in detail, the use of finite elements for solving problems in which plasticity effects are present. To our knowledge it is the first such book. Our main idea is to present the theory and detailed algorithms in the form of modular routines written in FORTRAN which can be linked together to form 13 finite element plasticity programs.

Writing this book has been in itself, rather like solving a nonlinear finite element problem. We have gone through many iterations and we hope that we have now converged to a reasonable 'solution'. As in many real engineering situations our convergence criterion has been influenced by a deadline. In our case the deadline was largely self-imposed as we have already been engaged on this project for more than three years. We do not believe our solution to be unique or in any sense optimal. We merely offer it to fill a gap in the existing literature.

The text is arranged in three main parts. Part I is devoted to one-dimensional problems. These relatively simple applications are possibly the most important in the book; since all the essential features of nonlinear finite element analysis are immediately recognisable without the distractions and complications that are present in general continuum problems. Part II deals with the two-dimensional applications of plane stress/strain and axisymmetric continua and plate bending problems. Finally in Part III we present some dynamic transient applications and briefly describe some further developments.

All of the programs presented in this text have been specially written by the authors. In the development of the subroutines for the solution algorithms described, a conflict inevitably arose between computational efficiency and clarity of coding. Whatever sacrifices have been made have been biased towards satisfying the latter condition. However, we believe that the codes presented are both reasonably efficient and flexible and have potential usage in commercial as well as teaching and research environments. A total of 132 subroutines are presented which amount to more than 8,000 statements. The 13 assembled programs comprise approximately 20,000 statements. To aid readers wishing to implement the programs a magnetic tape of the computer codes together with the test input data listed in Appendix IV is available from the publishers. Although every attempt has been made to verify the programs, no responsibility can be accepted for their performance in practice.

A further feature of the book is that each chapter contains several exercises for further study.

We are indebted to many people for their direct or indirect assistance in the preparation of this text. This preface would not be complete without an acknowledgment of this debt and a record of our gratitude to the following: To Professor O. C. Zienkiewicz for his pioneering work and stimulating influence. To Professor G. C. Nayak whose work on numerical analysis of plasticity problems has significantly influenced the present text. To Dr. I. C. Cormeau whose thesis on viscoplasticity has been an invaluable source of information. To Professor K. J. Bathe for permission to use the profile equation solver included in Chapter 11. To N. Bicanic, D. K. Paul, H. H. Abdel Rahman and M. M. Huq for their generous assistance in the preparation of several chapters. To our colleagues and former research workers in the Department of Civil Engineering, University College of Swansea for helpful discussions and suggestions. To E. S. Caldis for his care in preparing annotated computer listings and, finally, to Mrs. M. J. Davies for her skill and patience in typing the manuscript.

D. R. J. OWEN
E. HINTON

Swansea, May 1980

Part I

Chapter 1
Introduction

1.1 Introductory remarks

The finite element method is now firmly accepted as a most powerful general technique for the numerical solution of a variety of problems encountered in engineering. Applications range from the stress analysis of solids to the solution of acoustical, neutron physics and fluid dynamics problems. Indeed the finite element process is now established as a general numerical method for the solution of partial differential equation systems, subject to known boundary and/or initial conditions.

For linear analysis, at least, the technique is widely employed as a design tool. Similar acceptance for nonlinear situations is dependent on two major factors. Firstly, in view of the increased numerical operations associated with nonlinear problems, considerable computing power is required. Developments in the last decade or so have ensured that high-speed digital computers which meet this need are now available and present indications are that reductions in unit computing costs will continue. Secondly, before the finite element method can be used in design, the accuracy of any proposed solution technique must be proven. The development of improved element characteristics and more efficient nonlinear solution algorithms and the experience gained in their application to engineering problems have ensured that nonlinear finite element analyses can now be performed with some confidence. Hence barriers to the common use of nonlinear finite element techniques are being rapidly removed and the process is already economically acceptable for selected industrial applications.

1.2 Aims and layout

The object of this book is to describe in detail the application of the finite element method to the solution of materially nonlinear engineering analysis problems. Unlike other texts on linear and nonlinear finite element analysis[1–4] which have dealt predominantly with theoretical aspects, this book is intended to be more practical and therefore focuses attention on the *computer implementation* of nonlinear finite element schemes.

Nonlinearities arise in engineering situations from several sources. For example a nonlinear material response can result from elasto-plastic material behaviour or from hyperelastic effects of some form. Additionally nonlinear characteristics can be associated with temporal effects such as viscoplastic behaviour or dynamic transient phenomena. Each of these nonlinearities may occur in a variety of structural types such as two- or three-dimensional solids, frames, plates or shells. Therefore it becomes clear that a textbook dealing with nonlinear finite element programming must at least be restricted to selected topics. For this reason three classes of problems will be examined in depth in the three parts of this text.

Part I: One-dimensional materially nonlinear problems. All the essential features of a nonlinear finite element solution can be described in relation to one-dimensional models. The applications considered are:

- Nonlinear quasi-harmonic problems
- Nonlinear elastic situations
- Elasto-plastic behaviour of axial bar systems
- Time dependent elasto-viscoplastic analysis of bar systems
- Elasto-plastic beam bending

Part II: Two-dimensional materially nonlinear problems. In this part the ideas developed in Part I are extended to continuum problems. The following applications are presented:

- Elasto-plastic analysis of plane stress, plane strain and axi-symmetric solids
- Time dependent elasto-viscoplastic analysis of plane stress, plane strain and axisymmetric solids
- Elasto-plastic plate bending problems

Part III: Nonlinear transient dynamic problems. In this time-dependent class of problems inertia effects are included in the analysis. In this part, the following topics are considered:

- Elasto-plastic and geometrically nonlinear material behaviour
- Explicit and implicit time integration schemes
- Combined explicit/implicit algorithms

It should be pointed out that several different programming options are open for solution of the above problems and the methods presented in this text are the ones which are physically the most clear and which experience indicates give reliable results for a wide range of applications. An important feature of this text is the step-by-step development of thirteen finite element programs to deal with the above problems.

For the one-dimensional applications considered in Part I, only a 2-node element with linear displacement variation between nodes is considered. This allows the basic steps of a nonlinear finite element analysis to be presented without unnecessary distractions. In Parts II and III of the text, where two-dimensional continuum and plate bending problems are considered, isoparametric elements are exclusively employed. In particular, a

4-node linear element and 8- and 9-node quadratic versions are used. These elements are illustrated in Fig. 1.1 and are extremely versatile, good performers which have been well tried and tested in both linear and nonlinear situations. A typical elasto-plastic application using 8-node isoparametric elements is shown in Fig. 1.2 where the incremental loading of a notched beam is illustrated. The progressive development of plastic zones with increasing load levels are compared for a Tresca and Von Mises yield criterion.

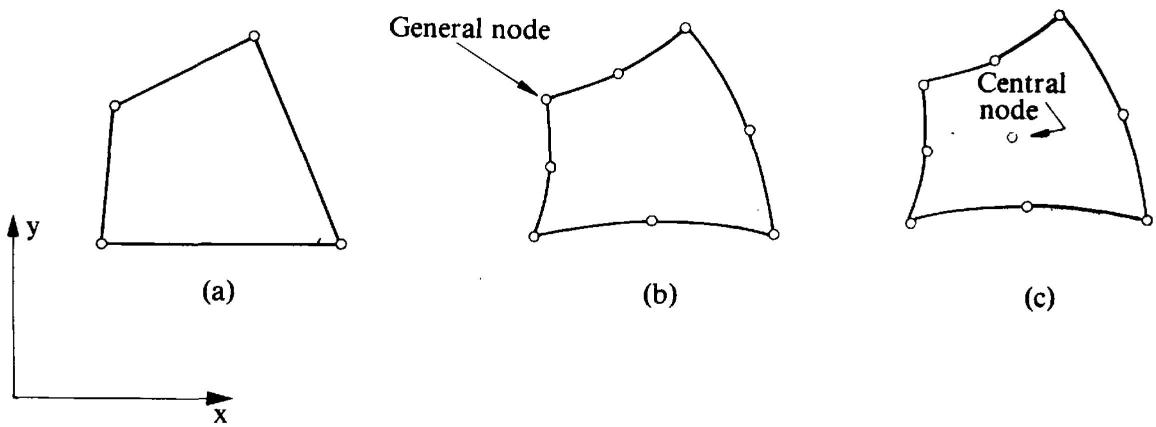


Fig. 1.1 The two-dimensional isoparametric elements employed in the text: (a) Linear 4-node; (b) Serendipity 8-node; (c) Lagrangian 9-node.

The layout of the book will now be briefly described. The remainder of Chapter 1 discusses the basic notation and style adopted in program presentation.

Chapter 2 discusses the general nonlinear problem and some solution techniques are outlined. For the one-dimensional applications to be considered, basic theoretical expressions are developed in a form suitable for numerical solution.

In Chapter 3, the solution techniques presented in Chapter 2 are programmed in FORTRAN and numerical examples are solved for each separate application.

Chapter 4 is devoted to one-dimensional elasto-viscoplastic problems. The basic theory for this time-dependent phenomenon is first presented. The process is then coded and the program used to solve some numerical examples.

In Chapter 5 elasto-plastic beam bending is considered. This topic forms a bridge between uniaxial and continuum applications since now more than one degree of freedom exists at each nodal point. Some measure of continuum behaviour is also introduced since a layered approach is used to trace the development of plasticity through the cross-section of the beam.

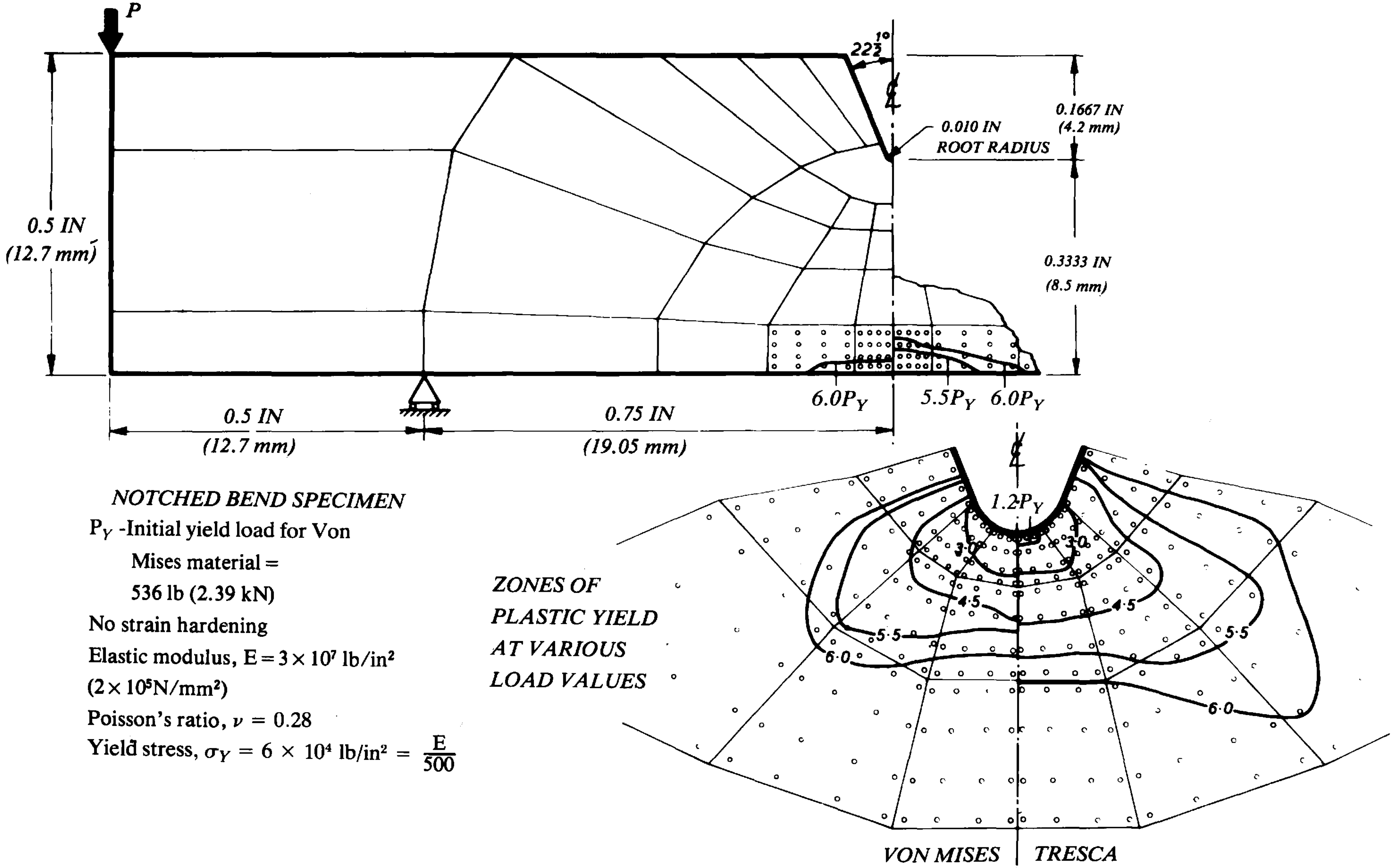


Fig. 1.2 Elasto-plastic analysis of a notched beam under bending showing plastic zone distributions for both a Von Mises and a Tresca yield criterion.

Chapter 6 forms an introduction to two-dimensional continuum problems. The basic theory for two-dimensional isoparametric elements is presented and some standard subroutines required for applications described in later chapters are listed. These include routines which perform some standard linear elastic operations, such as nodal load generation, equation solution, etc., as well as nonlinear subroutines common to more than one application.

Two-dimensional elasto-plastic problems are considered in Chapter 7. Basic theoretical expressions for a general continuum are first reviewed, and manipulated into forms convenient for numerical analysis. Particular expressions for plane stress/strain and axisymmetric situations are then developed and coded. Four different yield criteria are employed. The Tresca and Von Mises laws which closely approximate metal plasticity behaviour are considered and the Mohr–Coulomb and Drucker–Prager criteria, which are applicable to concrete, rocks and soil are presented.

Chapter 8 is concerned with the transient phenomenon of elasto-viscoplasticity where again the situations of plane stress/strain and axial symmetry are considered. Both explicit and implicit time integration schemes are presented and the four yield criteria considered in Chapter 7 are employed. The FORTRAN program developed is illustrated by application to some numerical examples.

Elasto-plastic plate bending problems are discussed in Chapter 9. The basic theoretical expressions are presented in a form suitable for numerical analysis with both a layered and nonlayered approach to plastification through the plate thickness being considered. Treatment in this chapter is limited to the Tresca and Von Mises yield conditions.

Chapters 10 and 11 deal with the transient dynamic analysis of two-dimensional continua. In this application inertia effects are included in the computation and problems such as blast loading and seismic phenomena are considered. Nonlinear effects due to both elasto-plastic material behaviour and gross geometric deformations are included. Both explicit and implicit techniques are employed for the time integration of the equations of motion as well as a combined implicit/explicit algorithm. The computer codes developed are applied to the solution of some practical problems.

Finally in Chapter 12 further aspects of nonlinear material behaviour are discussed. Alternative solution techniques and material models are referred to and some additional fields of application indicated.

Three appendices are included which contain user instructions for the computer programs described throughout the text. Appendices I and II provide user instructions for one-dimensional and two-dimensional continuum problems respectively. A user's guide for transient dynamic problems is provided in Appendix III. Finally in Appendix IV sample input data and lineprinter output are provided for both one- and two-dimensional applications.

1.3 Program structure

1.3.1 Introduction

This section describes the main features of the computer programs to be developed later in the book. A modular approach is adopted, in that separate subroutines are employed to perform the various operations required in a nonlinear finite element analysis. Generally each program consists of 9 modules, each with a distinct operational function. Each module in turn is composed of one or more subroutines relevant only to its own needs and, in some cases, of subroutines which are common to several modules. Control of the modules is held by the main or master segment.

The modules, shown schematically in Fig. 1.3, are described in relation to their general functions as follows:

1. *Initialisation or zeroing module*—this is the first module entered and its function is to initialise to zero various vectors and matrices at the beginning of the solution process.
2. *Data input and checking module*—this is the second module entered. It handles input data defining the geometry, boundary conditions and material properties. This data is checked using diagnostic routines and if errors occur they are flagged and the remainder of the input data is printed out before the program is terminated. For isoparametric elements, Gaussian integration constants and mid-side nodal coordinates for straight-sided elements are also evaluated in this section. Once used this module is not needed again.
3. *Loading module*—this module organises the calculation of nodal forces due to the various forms of loading for two-dimensional application. These include pressure, gravity and concentrated loadings.
4. *Load incrementing module*—Any materially nonlinear finite element solution must proceed on an incremental basis. Therefore the function of this section is to control the incrementing of the applied loads evaluated by the loading module. It also ensures that any specified displacement values are also incrementally applied.
5. *Stiffness module*—this is the next module entered and organises the evaluation of the stiffness matrix for each element. The stiffness matrices are stored on disc and ordered in the sequence required for equation assembly and reduction.
6. *Solution module*—the general purpose of this routine is to assemble, reduce and solve the governing set of simultaneous equations to give the nodal displacements and force reactions at restrained nodal points.
7. *Residual force module*—the function of this module is to calculate the residual or 'out of balance' nodal forces at each stage of the analysis.
8. *Convergence module*—in this module the convergence of the nonlinear solution is checked against criteria given in later chapters.

9. *Output module*—this module organises the output of the requested quantities.

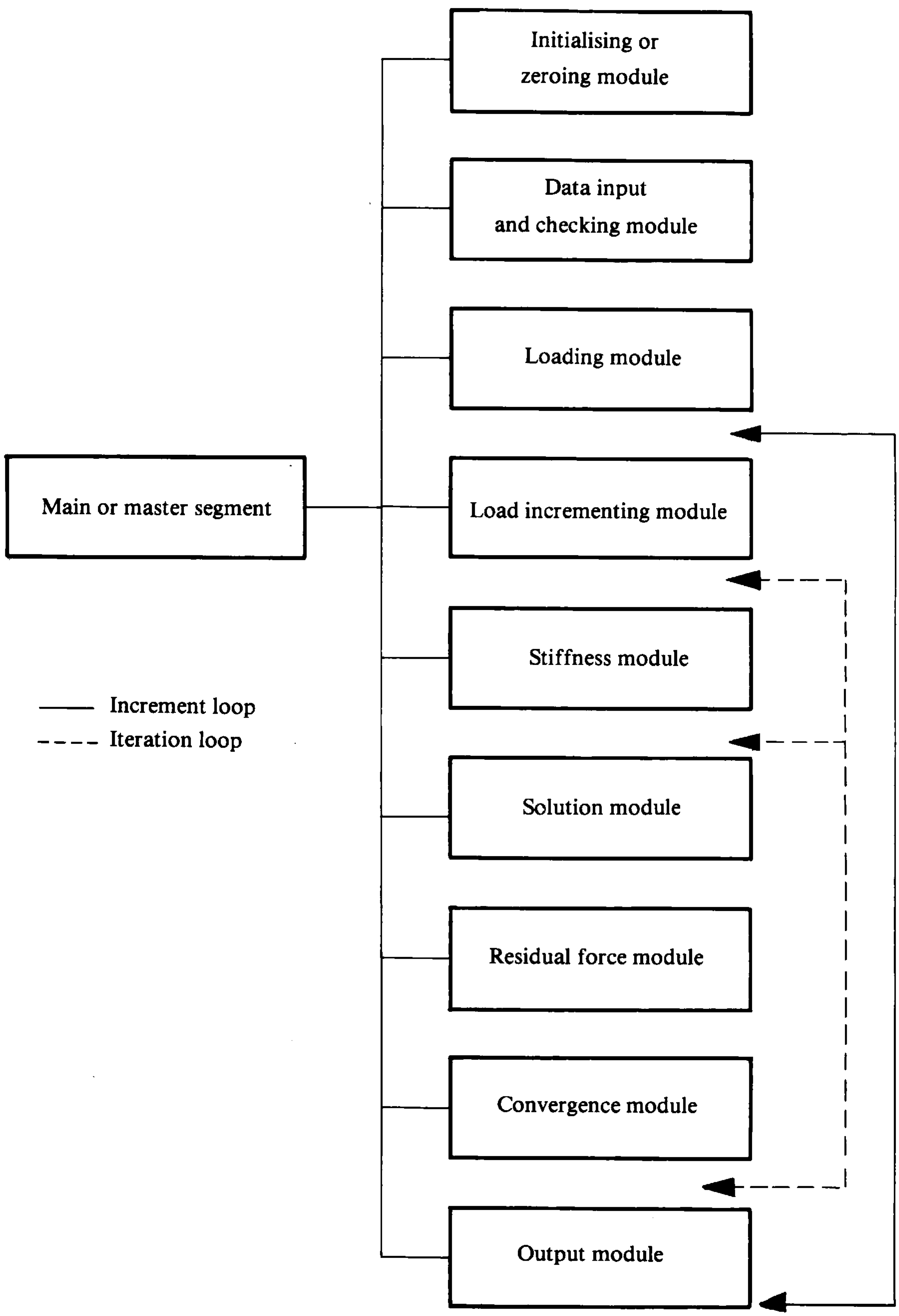


Fig. 1.3 Program modules for nonlinear solution codes.

The main purpose of the main or master segment is to call the above modules and to control the load increments and iteration procedure according to the solution algorithm being employed and the convergence rate of the solution process.

1.3.2 Programming notation

In the programs presented in this text an attempt has been made to name variables in a logical manner. By choosing descriptive names, the use of many of the variables becomes self-apparent, thus assisting the reader in the task of program assimilation. All variable names are chosen to be 5 characters in length; this occasionally causes a little difficulty in abbreviation but has an advantage with regard to neatness of program presentation. For example, the following names will be employed.

NMATS	The Number of different MATerialS
PROPS ( )	The array of material PROPertieS
NEVAB	The Number of Element VAriaBles
NNODE	The Number of NODes per Element
NDOFN	The Number of Degrees Of Freedom per Node

Furthermore a 'common root' principle will be adopted; where a single basic variable name is employed with different prefixes depending on its usage in the program. In particular:

i) Prefix I, J or L will be used to indicate a DO loop variable
ii) Prefix K will indicate a counter
iii) Prefix M will indicate a maximum value
iv) Prefix N will indicate a given number

For example IPOIN, NPOIN, MPOIN will indicate respectively a particular nodal point, the number of nodal points in the problem and the maximum permissible number of nodal points in the program.

Similarly, any DO loop will be of the general form

```
  KEVAB=0
  DO 1 INODE=1, NNODE
  DO 1 IDOFN=1, NDOFN
1 KEVAB=KEVAB+1
```

which indicates that the outer and inner DO loop indices range respectively over the number of nodes per element and the number of degrees of freedom per node. The prefix K is employed in KEVAB to indicate a counter over the number of element variables, NEVAB.

All programming is undertaken in standard FORTRAN IV. A listing is presented for all subroutines described in this text and detailed notes on each group of statements are provided. Comment cards have also been used to assist in the understanding of the programs.

1.4 References

1. ZIENKIEWICZ, O. C. *The Finite Element Method*, McGraw-Hill, 1977.
2. ODEN, J. T., *Finite Elements of Nonlinear Continua*, McGraw-Hill, 1972.
3. DESAI, C. S. and ABEL, J. F., *An Introduction to the Finite Element Method*, Van Nostrand Reinhold, New York, 1972.
4. GALLAGHER, R. H., *Finite Element Analysis—Fundamentals*, Prentice Hall, 1975.
5. HINTON, E. and OWEN, D. R. J., *Finite Element Programming*, Academic Press, 1977.
6. HINTON, E. and OWEN, D. R. J., *An Introduction to Finite Element Computations*, Pineridge Press, Swansea, U.K., 1979.

Chapter 2
One-dimensional nonlinear problems

2.1 Introduction

Several classes of nonlinear problems of interest in many branches of science and engineering can be reduced to the solution of a system of simultaneous equations in which the equation coefficients are dependent on some function of the prime variables.(1) In this chapter some basic techniques for the numerical solution of such problems are examined. In order to introduce the essential details of the solution processes as simply as possible, the applications will be restricted to one-dimensional situations. In particular, elasto-plasticity, nonlinear elasticity problems and systems governed by a nonlinear quasi-harmonic equation will be considered. In each case a computer program will be developed and its use illustrated by application to simple problems. The aim of this chapter is to prepare the reader for the more comprehensive two-dimensional treatment of these topics which will be undertaken in Chapters 6–9. Indeed, all the essential features of nonlinear finite element analysis detailed in these later chapters will be recognisable from the simple treatment considered here. It should be emphasised that the subroutines developed in this chapter will *not* be used in the main finite element programs discussed in Parts II and III.

2.2 Basic numerical solution processes for nonlinear problems

The use of finite element discretisation in a large class of nonlinear problems results in a system of simultaneous equations of the form

$$\boldsymbol{H}\boldsymbol{\varphi}+\boldsymbol{f} = 0, \tag{2.1}$$

in which $\boldsymbol{\varphi}$ is the vector of the basic unknowns, $\boldsymbol{f}$ is the vector of applied 'loads' and $\boldsymbol{H}$ is the assembled 'stiffness' matrix. For structural applications, the terms 'load' and 'stiffness' are directly applicable, but for other situations the interpretation of these quantities varies according to the physical problem under consideration.

If the coefficients of the matrix $\boldsymbol{H}$ depend on the unknowns $\boldsymbol{\varphi}$ or their derivatives, the problem clearly becomes nonlinear. In this case, direct solution of equation system (2.1) is generally impossible and an iterative scheme must be adopted. Many options remain open for the iterative sequence to be employed. Some of the most generally applicable methods available will now be outlined.

2.2.1 Method of direct iteration (or successive approximations)

In this approach[2] successive solutions are performed, in each of which the previous solution for the unknowns $\boldsymbol{\varphi}$ is used to predict the current values of the coefficient matrix $\boldsymbol{H}(\boldsymbol{\varphi})$. Rewriting (2.1) as

$$\boldsymbol{\varphi} = -[\boldsymbol{H}(\boldsymbol{\varphi})]^{-1}\boldsymbol{f}, \tag{2.2}$$

then the iterative process yields the $(r+1)^{\text{th}}$ approximation to be

$$\boldsymbol{\varphi}^{r+1} = -[\boldsymbol{H}(\boldsymbol{\varphi}^r)]^{-1}\boldsymbol{f}. \tag{2.3}$$

If the process is convergent then in the limit as r tends to infinity $\boldsymbol{\varphi}^r$ tends to the true solution.

It is seen from (2.3) that it is necessary to recalculate the 'stiffness' matrix $\boldsymbol{H}$ for each iteration. To commence the process, an initial guess for the unknown $\boldsymbol{\varphi}$ is required in order to calculate $\boldsymbol{H}$. Generally a value of $\boldsymbol{\varphi}^0$ based on the solution for an average material property throughout the region is found to be satisfactory. If the nonlinearity of the material properties is very marked at certain values of $\boldsymbol{\varphi}$, an approximate prescription of the field variable at all nodes may be necessary.

For practical purposes, the iterative process is deemed to have converged when some measure (usually a norm of the nodal unknowns) of the change in the unknown $\boldsymbol{\varphi}$ between successive iterations has become tolerably small. The process is illustrated diagrammatically for a single variable in Figs 2.1 and 2.2, in which case the matrix $\boldsymbol{H}$ and vector $\boldsymbol{\varphi}$ reduce to the scalar equivalents H and ϕ. The assumed dependence of H on ϕ is a basic problem function which must be prescribed before solution can commence. This material property is included in Figs 2.1 and 2.2 and, for convenience, the relationship between $H(\phi).\phi$ and ϕ is prescribed rather than the $H(\phi)-\phi$ dependence. Figure 2.1 shows the convergence paths for initial trial values, ϕ^0, which are below and above the true solution, ϕ_T, and for a convex $H-\phi$ relation. From the initial trial value, ϕ^0, the corresponding value of H is immediately given from the prescribed $H(\phi).\phi-\phi$ relationship, to be H^0. Equation (2.3) is then solved to give ϕ^1. The value of H corresponding to ϕ^1 is then determined from the $H(\phi).\phi-\phi$ relationship and (2.3) then resolved to obtain ϕ^2. This cycling process is continued until ϕ^{n-1} and ϕ^n are deemed to be sufficiently close, indicating that convergence has occurred. The quantity H^r is represented by the slope of the secant to the $H-\phi$ curve and decreases with increasing values of ϕ. Both the high and low initial trial solutions produce monotonic convergence paths. Figure 2.2 shows the unsuitability of the method for problems with a concave $H-\phi$ relationship. Both low and high initial trial solutions produce convergence paths which oscillate around the true solution. Although the solution converges for the

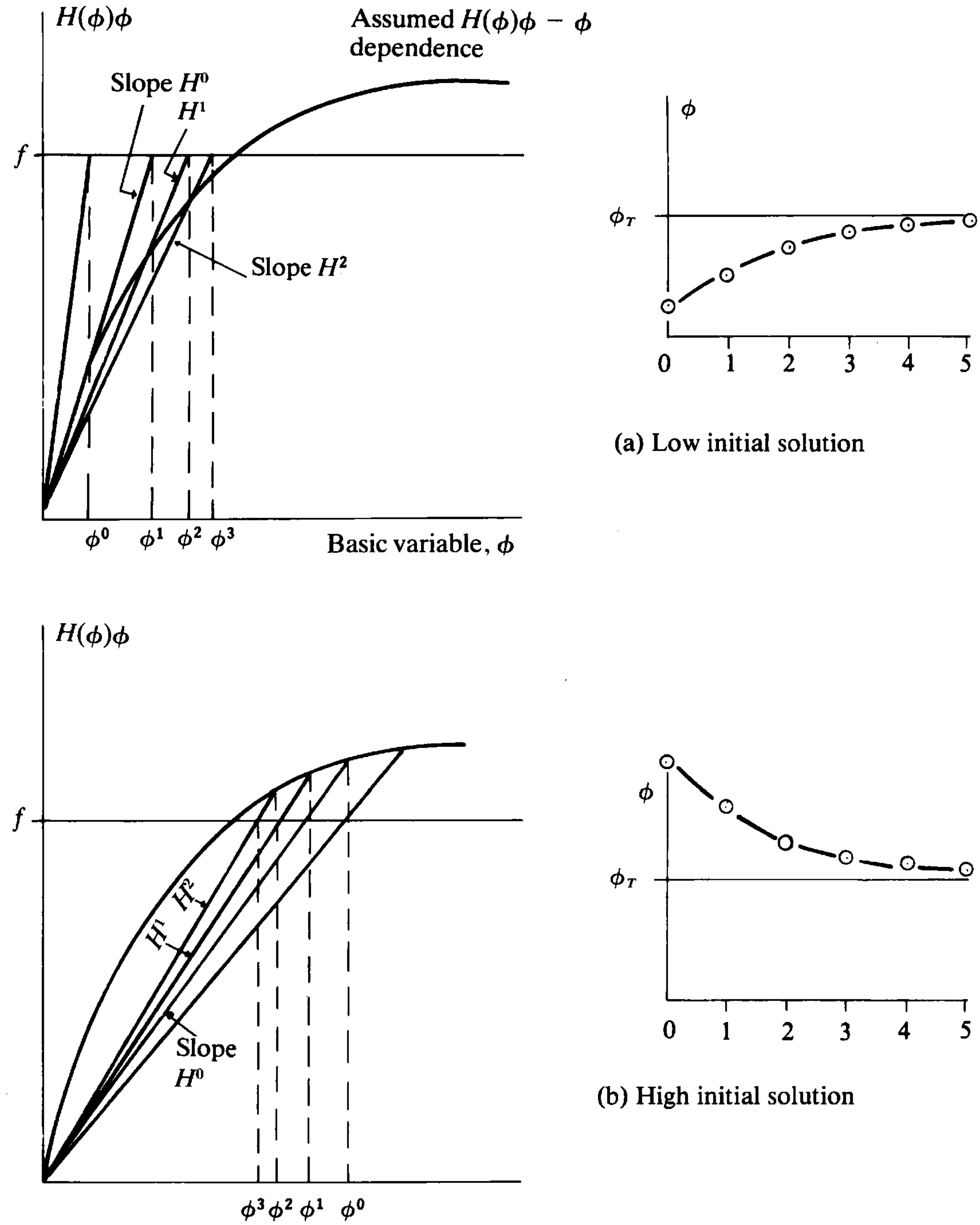


Fig. 2.1 Direct iteration method for a single variable problem—convex H–ϕ relation.

single variable case, in multi-degree of freedom problems the coupling of stiffness terms is likely to lead to instability of the iterative process. A disadvantage of the direct iteration method is that convergence of the solution scheme is not guaranteed and cannot be predicted at the initial solution stage.

2.2.2 The Newton–Raphson method

During any step of an iterative process of solution, (2.1) will not be satisfied unless convergence has occurred. A system of *residual forces* can be assumed

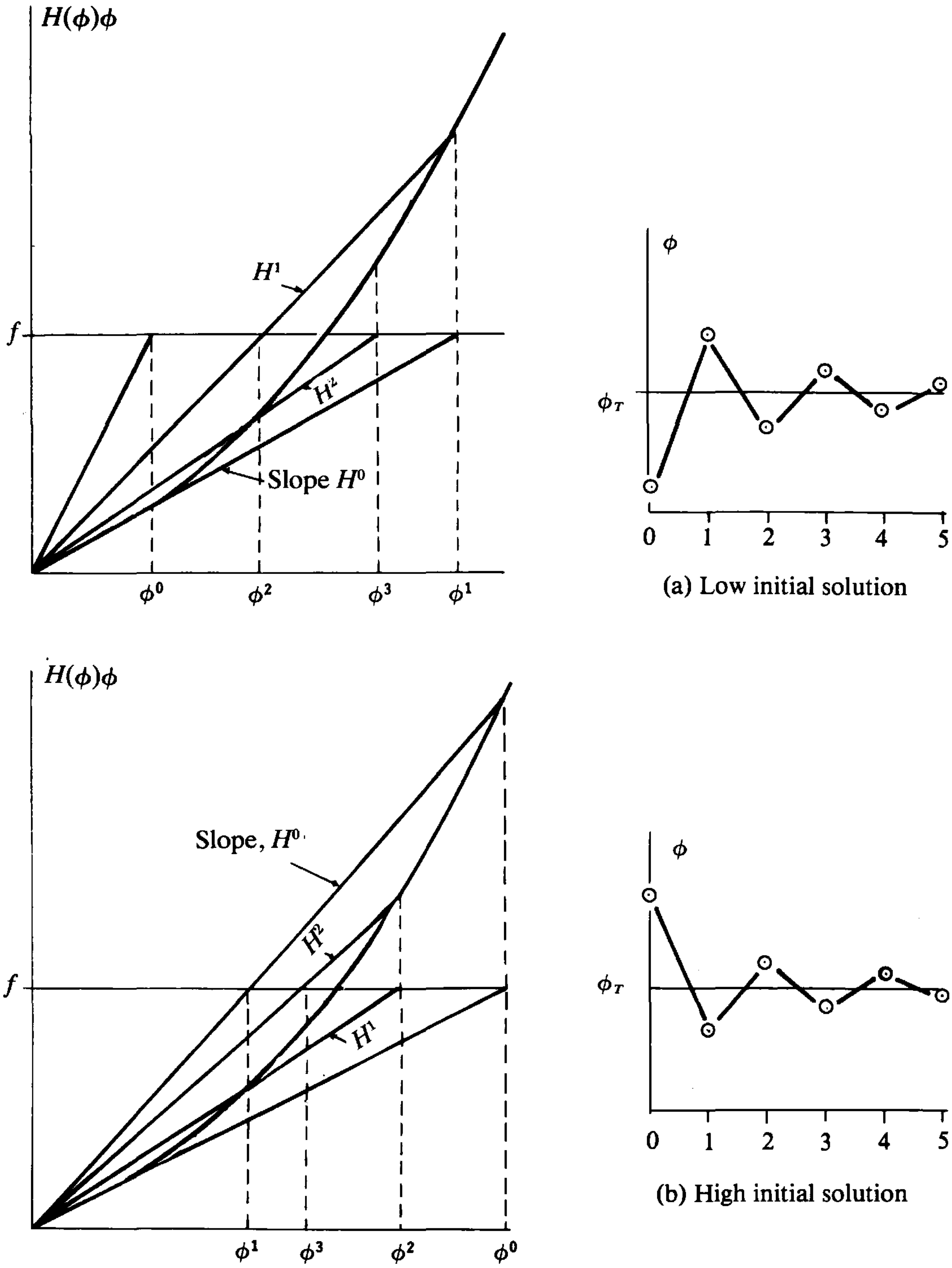


Fig. 2.2 Direct iteration method for a single variable problem—concave H–ϕ relation.

to exist, so that

$$\psi = H\varphi + f \neq 0. \tag{2.4}$$

These residual forces ψ can be interpreted as a measure of the departure of (2.1) from equilibrium. Since H is a function of φ and possibly its derivatives, then at any stage of the process, $\psi = \psi(\varphi)$.

If the true solution to the problem exists at $\boldsymbol{\varphi}^r+\Delta\boldsymbol{\varphi}^r$ then the Newton–Raphson approximation[2] for the general term of the residual force vector, $\boldsymbol{\psi}^r$ corresponding to solution at $\boldsymbol{\varphi}^r$ is

$$\psi_i{}^r = -\sum_{j=1}^{N} \Delta\phi_j{}^r \left(\frac{\partial\psi_i}{\partial\phi_j}\right)^r, \tag{2.5}$$

in which N is the total number of variables in the system and the superscript r denotes the r^{th} approximation to the true solution. Substituting for ψ_i from (2.4), the complete expression for all the residual components can be written in matrix form as

$$\boldsymbol{\psi}(\boldsymbol{\varphi}^r) = -\boldsymbol{J}(\boldsymbol{\varphi}^r)\Delta\boldsymbol{\varphi}^r. \tag{2.6}$$

in which a typical term of the Jacobian matrix $\boldsymbol{J}$ is

$$J_{ij} = \left(\frac{\partial\psi_i}{\partial\phi_j}\right)^r = h_{ij}{}^r + \sum_{k=1}^{m}\left(\frac{\partial h_{ik}}{\partial\phi_j}\right)^r \phi_k{}^r, \tag{2.7}$$

where h_{ij} is the general term of matrix $\boldsymbol{H}$. The last term in (2.7) gives rise to nonsymmetric terms in the Jacobian matrix. If these nonsymmetric terms are neglected in order to maintain symmetry, then substitution of (2.7) in (2.6) results in

$$\boldsymbol{H}(\boldsymbol{\varphi}^r).\Delta\boldsymbol{\varphi}^r = -\boldsymbol{\psi}(\boldsymbol{\varphi}^r). \tag{2.8}$$

Or since

$$\Delta\boldsymbol{\varphi}^r = \boldsymbol{\varphi}^{r+1}-\boldsymbol{\varphi}^r, \tag{2.9}$$

equation (2.8) reduces, on use of (2.4), to

$$\boldsymbol{H}(\boldsymbol{\varphi}^r).\boldsymbol{\varphi}^{r+1}+\boldsymbol{f} = 0. \tag{2.10}$$

This equation is identical to equation (2.3), Section 2.2.1, which governs the method of direct iteration. Therefore in order to achieve the better convergence rate associated with the Newton–Raphson process it is essential that the unsymmetric terms in $\boldsymbol{J}$ be retained.

The explicit form of the nonlinear terms in (2.7) will clearly depend on the way in which the stiffness matrix coefficients, h_{ij}, depend on the unknowns, $\boldsymbol{\varphi}$. The terms of the Jacobian matrix, given in (2.7), can be assembled to give the general expression

$$\boldsymbol{J}(\boldsymbol{\varphi}) = \boldsymbol{H}(\boldsymbol{\varphi})+\boldsymbol{H}'(\boldsymbol{\varphi}), \tag{2.11}$$

where the last term contains the unsymmetric terms only. The Newton–Raphson process can be finally written, using (2.6) and (2.11), in the form

$$\Delta\boldsymbol{\varphi}^r = -[\boldsymbol{J}(\boldsymbol{\varphi}^r)]^{-1}.\boldsymbol{\psi}(\boldsymbol{\varphi}^r) = -[\boldsymbol{H}(\boldsymbol{\varphi}^r)+\boldsymbol{H}'(\boldsymbol{\varphi}^r)]^{-1}\boldsymbol{\psi}(\boldsymbol{\varphi}^r). \tag{2.12}$$

This allows the correction to the vector of unknowns φ to be obtained from the residual force vector ψ for any iteration. Again an iterative approach must be followed, with the vector of unknowns φ being corrected at each stage according to (2.12) until convergence of the process is deemed to have occurred. The technique is illustrated schematically in Figs 2.3 and 2.4 for

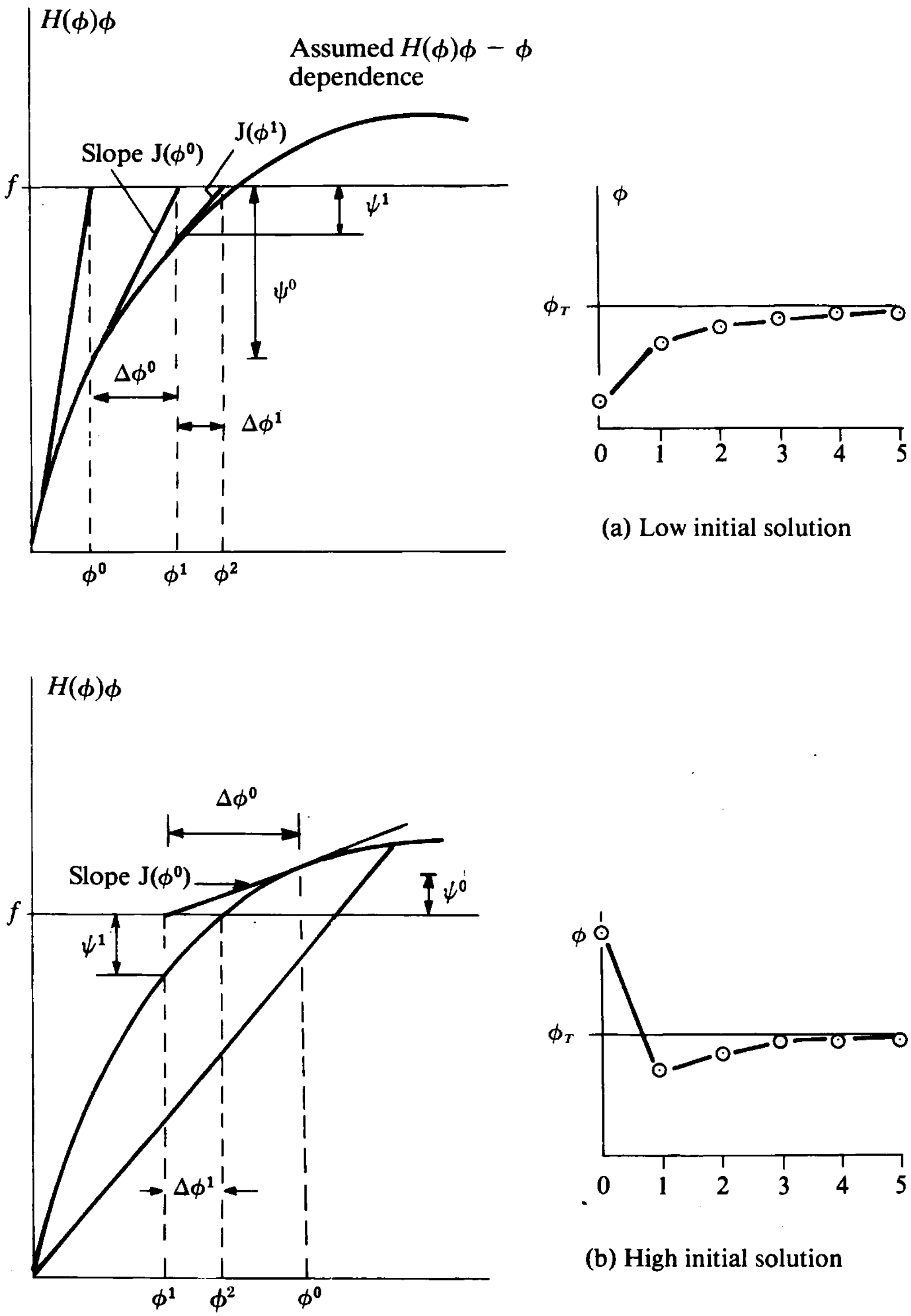


Fig. 2.3 The Newton–Raphson method for a single variable problem—convex H–ϕ relation.

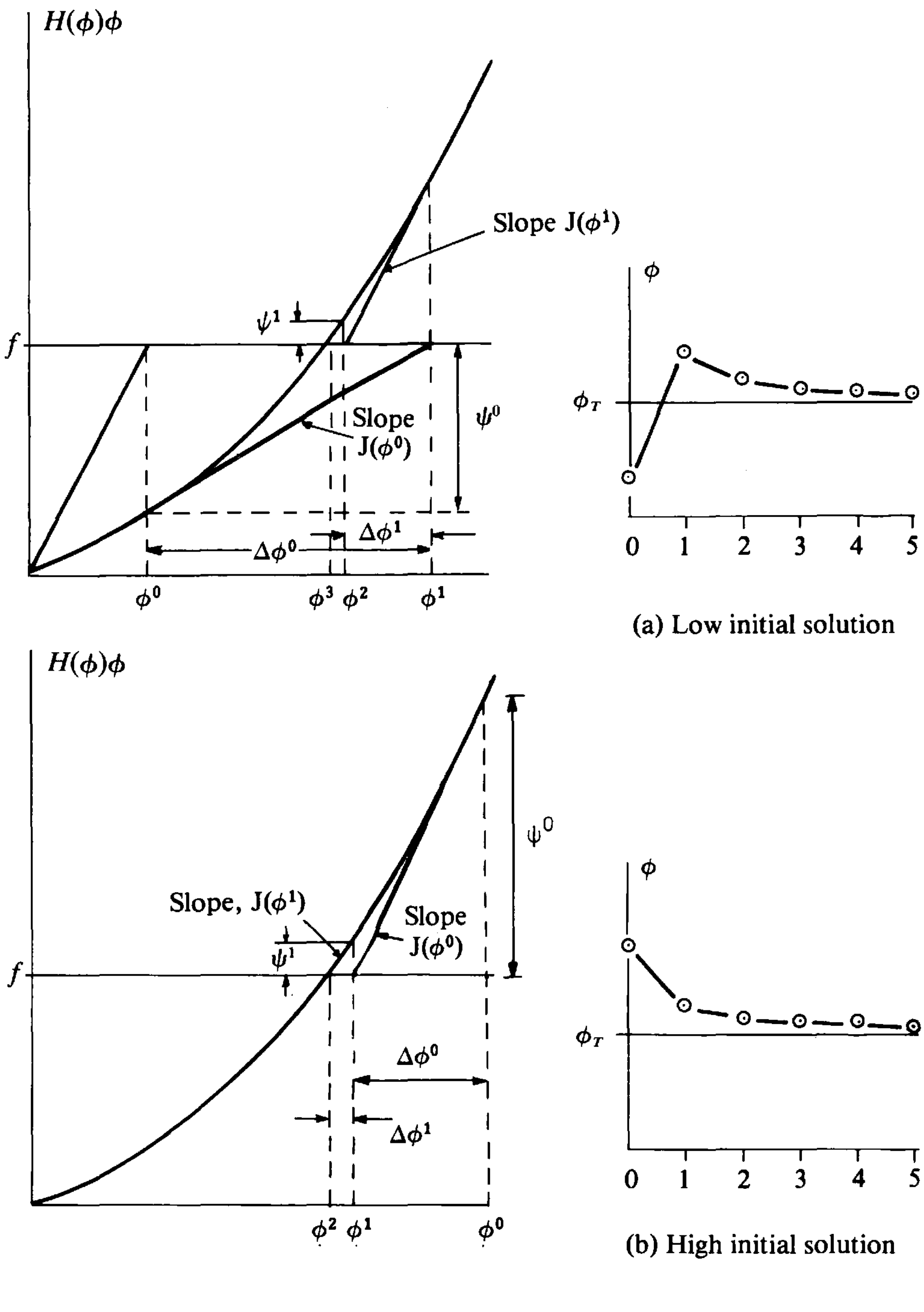


Fig. 2.4 The Newton–Raphson method for a single variable problem—concave H–ϕ relation.

a single variable situation. Solution to the nonlinear problem will be achieved when the residual force ψ vanishes, since this term directly measures the lack of equilibrium of the governing equation as indicated in (2.4). A trial value φ^0 of the basic unknown is assumed and the material stiffness associated with this value calculated according to the prescribed $H-\varphi$ relationship.

The residual force, ψ^0 is then calculated from (2.4) and the Jacobian evaluated according to (2.7). The correction $\Delta\varphi^0$ to the first approximation for the basic unknown, can finally be found from (2.12). Thus an improved approximation to the solution has been found, as $\varphi^1 = \varphi^0+\Delta\varphi^0$. This process can then be continually repeated until the residual force, ψ^n, is sufficiently small; or equivalently that φ^{r-1} and φ^r are sufficiently close. The Newton–Raphson process generally gives a more rapid and stable convergence path than the direct iteration method.

2.2.3 The tangential stiffness method

For structural applications the matrix $\boldsymbol{H}$ can be interpreted physically as the stiffness matrix of the structure. For nonlinear situations, in which the stiffness depends on the degree of displacement in some manner, $\boldsymbol{H}$ is equal to the local gradient of the force/displacement relationship of the structure at any point and is termed the tangential stiffness. The analysis of such problems must proceed in an incremental manner since the solution at any stage may not only depend on the current displacements of the structure, but also on the previous loading history. Consequently the problem can be linearised over any increment of load and therefore the matrix, which contains the nonlinear terms, can be discarded from (2.11) and (2.12). With this modification, the solution process is identical to that described in the previous section and for this reason the method is sometimes termed a generalised Newton–Raphson method.

The solution algorithm is illustrated in Fig. 2.5; again for a single variable situation. Solution is commenced from a trial value φ^0 of the unknown (for structural problems the starting position of solution is almost invariably $\varphi^0 = \mathbf{0}$). The tangential stiffness, $\boldsymbol{H}(\varphi^0)$, corresponding to this displacement state is then determined and the residual force ψ^0 calculated according to (2.4). The correction, $\Delta\varphi^0$, to the trial value is computed according to the linearised form of (2.12), which is

$$\Delta\varphi^r = -[\boldsymbol{H}(\varphi^r)]^{-1}\cdot\psi(\varphi^r) \tag{2.13}$$

An improved approximation to the unknown is then obtained as $\varphi^1 = \varphi^0+\Delta\varphi^0$. This iterative process is then continued until the solution converges to the nonlinear solution which is indicated by the condition that ψ^r practically vanishes.

2.2.4 The initial stiffness method

In the methods described in the three previous sections, the complete factorisation (or reduction) and solution of the full set of simultaneous equations describing the discretised structure is essential for each iteration. For the method of direct iteration the equation solution indicated by (2.3) is necessary, whilst the Newton–Raphson technique and tangential stiffness method demand the equation solutions indicated by (2.12) and (2.13)

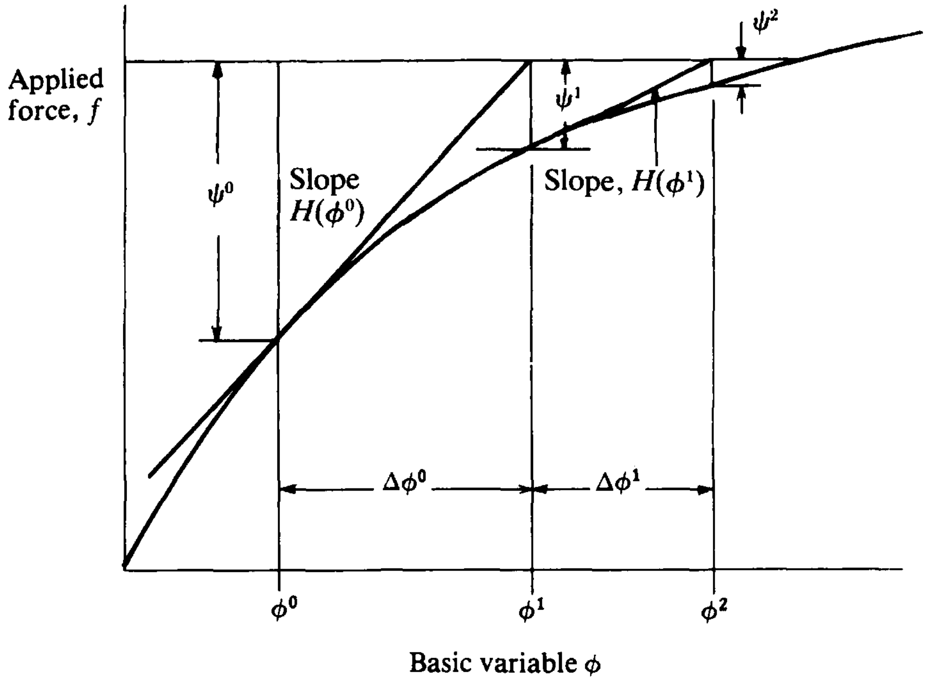


Fig. 2.5 Tangential stiffness solution algorithm for a single variable situation.

respectively. If in (2.13) the tangential stiffness matrix is replaced, at all steps of the computation, by the stiffness corresponding to the initial trial value of φ a complete factorisation, or reduction, of the assembled equations can be avoided.(3) In this case a complete equation solution need only be performed for the first iteration and subsequent approximations to the nonlinear solution performed, via the expression

$$\Delta\varphi^r = -[\boldsymbol{H}(\varphi^0)]^{-1}\psi(\varphi^r). \tag{2.14}$$

Since the same stiffness matrix $\boldsymbol{H}(\varphi^0)$ is employed at each stage, the reduced equations can be stored in their reduced or factored form and a second or subsequent solution merely necessitates the reduction of the right-hand side ($\psi(\varphi^r)$) terms, together with a backsubstitution. This has the immediate advantage of significantly reducing the computing cost per iteration but reduces the convergence rate as can be seen from Fig. 2.6 where the scheme is schematically illustrated. The iterative algorithm is identical to that described in the preceding section. This method can be shown to be unconditionally convergent(4) and can even be employed in situations where the material exhibits negative stiffness. The relative economies of the initial stiffness and tangential stiffness methods depend to a large extent on the degree of nonlinearity inherent in the problem under consideration. The optimum algorithm is generally provided by an amalgamation of both processes, in which the stiffnesses are changed at selected iterative intervals only.

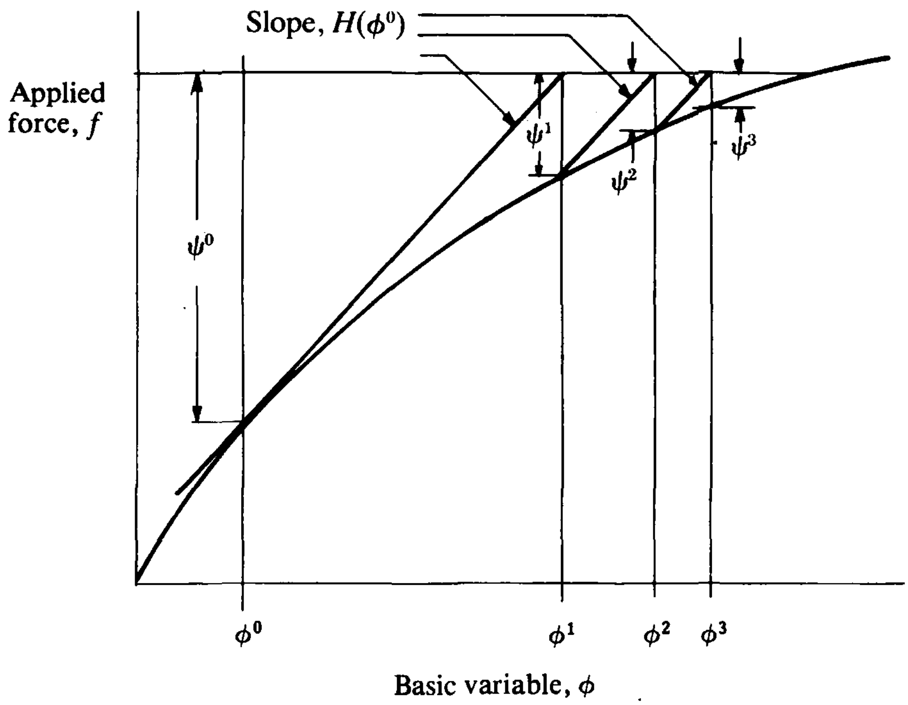


Fig. 2.6 Initial stiffness solution algorithm for a single variable situation.

2.3 Systems governed by a quasi-harmonic equation

Many physical situations in engineering science are governed by a quasi-harmonic equation containing coefficients which are dependent on the unknown variable or its derivatives according to some prescribed law. The most common problem of this type occurs in heat conduction under steady-state conditions when the material conductivity is itself a function of temperature. This phenomenon also arises in diffusion problems where the diffusivity of the medium often varies with the concentration of the diffusing matter. Further physical examples are provided in Ref. (5).

For a one-dimensional situation the governing equation to be considered is

$$\frac{d}{dx}\left(K\frac{d\phi}{dx}\right)+Q = 0, \tag{2.15}$$

in which ϕ is the unknown function and the terms K and Q may be functions of the position coordinate, x. The problem becomes nonlinear if K and/or Q are also functions of the unknown ϕ or its derivatives, according to some prescribed function.

Two types of boundary condition will be considered:

(a) The value of the unknown specified on the boundary

$$\phi = \phi_B. \tag{2.16}$$

(b) The gradient of the unknown at the boundary specified to be zero

$$\frac{d\phi}{dn} = \frac{d\phi}{dx} = 0. \tag{2.17}$$

(A more general form of this latter boundary condition is considered in Ref. 6.)

Equation (2.15) can be transformed to finite element form by suitable discretisation and use of the Galerkin weighted residual process.[5,6] The scalar product of equation (2.15) with any arbitrary weighting function, W, must be zero if ϕ satisfies (2.15) throughout any region Γ, so that

$$\int_{\Gamma}\left(\frac{d}{dx}\left(K\frac{d\phi}{dx}\right)+Q\right)W\,dx = 0. \tag{2.18}$$

Integrating the first term by parts results in

$$\left[WK\frac{d\phi}{dx}\right]_{x_1}^{x_2} - \int_{\Gamma}\left(K\frac{dW}{dx}\frac{d\phi}{dx} - QW\right)dx = 0, \tag{2.19}$$

where the limits of integration in the first term are the end points of the region Γ. The unknown function ϕ may be approximated as

$$\phi = \sum_{i=1}^{n} N_i\phi_i, \tag{2.20}$$

in which n is the total number of nodes in the finite element idealisation and N_i are the *global* shape functions. In the Galerkin process the number of weighting functions must equal the total number of unknown nodal values. The weighting function W_i corresponding to node i can then be conveniently chosen such that $W_i = N_i$. It should be noted that at nodes where the values of ϕ are prescribed, there is no associated unknown and consequently the weighting function for such nodes is zero. Therefore the first term in (2.19) always vanishes since at the two end points of the interval *either* ϕ is prescribed according to (2.16), in which case the weighting function for that point is zero, or $d\phi/dx$ is specified as zero according to (2.17). Substituting for ϕ and W in (2.19) and assembling all element contributions in the usual manner results in

$$\boldsymbol{H\varphi}+\boldsymbol{f} = \boldsymbol{0}, \tag{2.21}$$

in which typical element components are

$$h_{ij}{}^{(e)} = \int_{\Gamma^{(e)}} K\frac{dN_i{}^{(e)}}{dx}\frac{dN_j{}^{(e)}}{dx}\,dx, \tag{2.22}$$

$$f_i{}^{(e)} = \int_{\Gamma^{(e)}} QN_i{}^{(e)}\,dx, \tag{2.23}$$

where $N_i^{(e)}$ are the *element* shape functions specifying the distribution of the unknown, ϕ, over the element. For the specific case of a two-noded element with a linear variation in ϕ as shown in Fig. 2.7, the shape functions are simply

$$N_1^{(e)} = \frac{1}{2} - \frac{x}{L}, \qquad N_2^{(e)} = \frac{1}{2} + \frac{x}{L}, \tag{2.24}$$

where L is the length of the element.

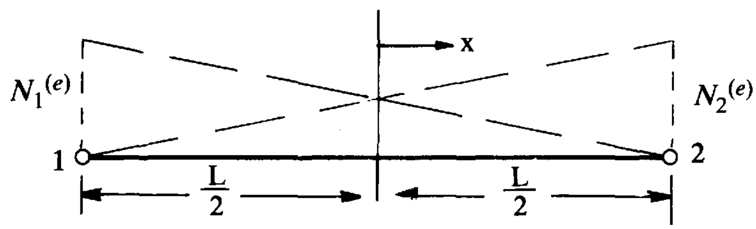


Fig. 2.7 One-dimensional two-noded element with linear variation of the unknown, ϕ, showing element shape functions.

Substituting in (2.22) and (2.23), and assuming no variation of K with position in the element, gives

$$\boldsymbol{H}^{(e)} = \frac{K}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \tag{2.25}$$

and

$$f_1^{(e)} = f_2^{(e)} = \frac{QL}{2}. \tag{2.26}$$

Provided that the variation of K with ϕ or its derivatives is specified, the problem falls into the category discussed in the previous section and can be solved by either the method of direct iteration or the Newton–Raphson approach.

In the numerical examples considered later in this chapter a specific form of nonlinearity will be considered, namely

$$K = K_0(a + b\phi), \tag{2.27}$$

in which K_0 is a reference value and a and b are known constants. For solution by the Newton–Raphson process the Jacobian matrix can be considered to be the sum of symmetric and nonsymmetric components as indicated in (2.11). The symmetric part has already been calculated in (2.25) and the nonsymmetric contribution must now be calculated according to the last term in (2.7). From (2.7), (2.22) and (2.27) the general term is given as

$$h_{ij}' = \sum_{k=1}^{2}\left(\frac{\partial h_{ik}}{\partial \phi_j}\right)\phi_k = \sum_{k=1}^{2}\left\{\phi_k K_0 \int_{-L/2}^{L/2} \frac{\partial}{\partial \phi_j}[a+b\phi]\frac{dN_i^{(e)}}{dx}\frac{dN_k^{(e)}}{dx}dx\right\}. \quad (2.28)$$

Noting that ϕ is given by (2.20) and that the shape functions are given by (2.24), the evaluation of (2.28) results in

$$\boldsymbol{H}'^{(e)} = \frac{K_0 b}{2L}(\phi_1-\phi_2)\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}. \quad (2.29)$$

As expected, it is seen that the derivative matrix $\boldsymbol{H}'^{(e)}$ is unsymmetric.

2.4 Nonlinear elastic problems

The simplest case of nonlinear behaviour in structural problems arises from nonlinear elastic material action. The stress/strain relationship of the material is nonlinear but the material behaviour is elastic with all deformations and displacements recoverable on unloading. For example, this type of behaviour arises in *hyperelastic* problems[7] where the stresses are functions of a strain dependent material modulus.

The nonlinear constitutive relation may be specified, for a one-dimensional situation, as

$$\sigma = \frac{dW}{d\epsilon} = E_0 \, . \, g(\epsilon) \quad (2.30)$$

where σ is the stress, ϵ the strain and E_0 some reference value of the material modulus. The material performance will be nonlinear according to the form of the specified strain energy function, $W(\epsilon)$.

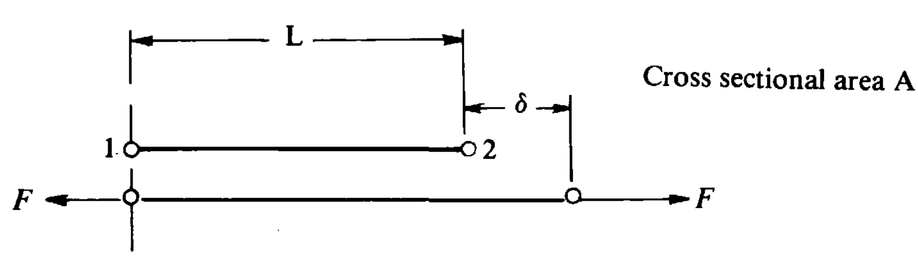


Fig. 2.8 Forces and displacements for a two-node element.

The simplest form of one-dimensional finite element is the constant stress element shown in Fig. 2.8 in which a linear displacement variation is assumed between nodes 1 and 2. The force in the element is given, from (2.30), by

$$F = E_0 A g(\delta/L), \quad (2.31)$$

where A is the element cross-sectional area and δ the element extension. The tangential stiffness for the material is then

$$K_T = \frac{dF}{d\delta} = \frac{E_0 A}{L}\frac{dg}{d\epsilon} = \frac{E_0 A}{L}g'(\epsilon). \tag{2.32}$$

Or, in particular, the element tangential stiffness matrix is given by

$$\boldsymbol{K_T}^{(e)} = \frac{E_0 A}{L}g'(\epsilon)\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{2.33}$$

Provided that $g'(\epsilon)$ is positive for all strain values, the tangential stiffness method of solution described in Section 2.2.3 can be employed in solution with $\boldsymbol{K_T}^{(e)}$ being directly equivalent to $\boldsymbol{H}(\boldsymbol{\varphi}^r)$. If the tangential stiffness matrix becomes zero, the assembled stiffness equations will become singular and the inversion process required by (2.13) cannot be undertaken. Solution for situations in which the material tangential stiffness becomes non-positive can be performed by use of the initial stiffness method described in Section 2.2.4. Since the initial material stiffness is employed throughout this latter process, the assembled stiffness matrix will remain positive definite throughout the computation.

2.5 Elasto-plastic problems in one dimension

In this section the essential features of elasto-plastic material behaviour are introduced, and the basic expressions are developed in a form suitable for numerical solution by some of the methods described in the previous sections.

Elasto-plastic behaviour is characterised by an initial elastic material response on to which a plastic deformation is superimposed after a certain level of stress has been reached.[8] Plastic deformation is essentially irreversible on unloading and is incompressible in nature. The onset of plastic deformation (or yielding) is governed by a *yield criterion* and post-yield deformation generally occurs at a greatly reduced material stiffness. Basic theoretical expressions for a general continuum are provided in Chapter 7.

For one-dimensional situations, the material parameters required to completely define elasto-plastic behaviour are most conveniently obtained from a uniaxial tension test. Figure 2.9 shows an idealised stress–strain curve for a material and identical behaviour is assumed in tension and compression. The material initially deforms according to the elastic modulus, E, until the stress level reaches a value σ_Y designated the *uniaxial yield stress*. On increasing the load further, the material is assumed to exhibit linear strain-hardening, characterised by the *tangential modulus*, E_T.

At some stage after initial yielding, consider a further load application resulting in an incremental increase of stress, $d\sigma$, accompanied by a change of strain, $d\epsilon$. Assuming that the strain can be separated into elastic and plastic

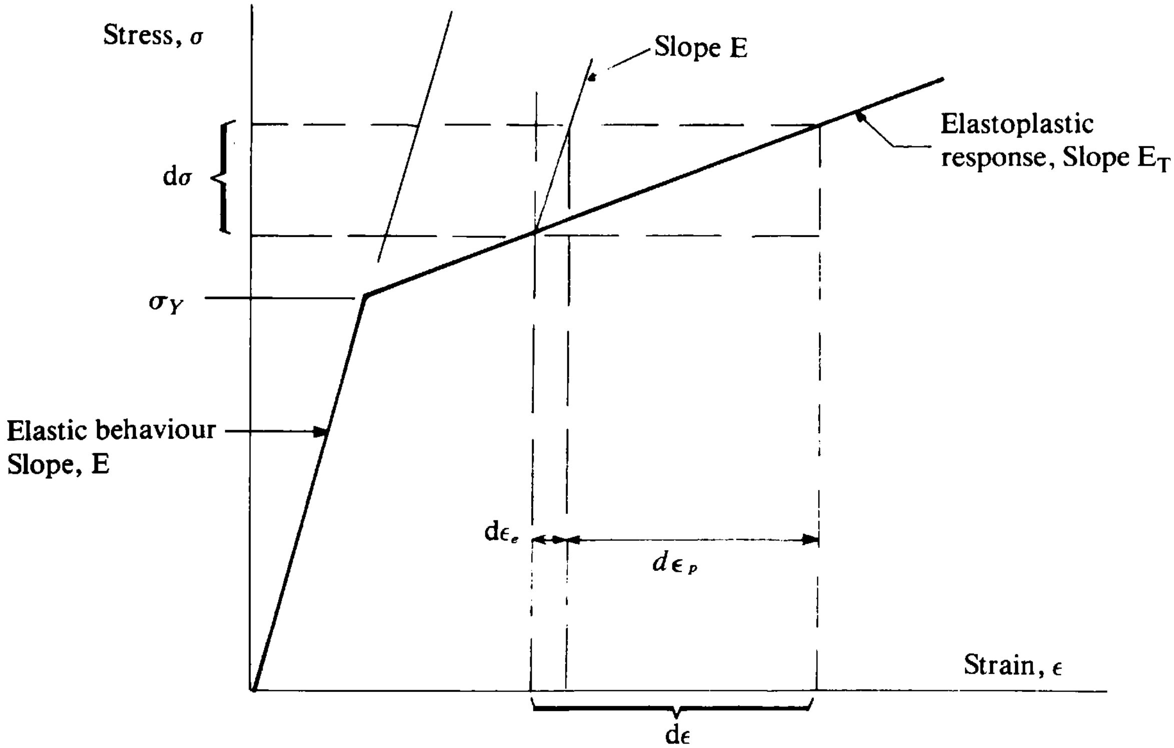


Fig. 2.9 Elastic, linear strain-hardening stress–strain behaviour for the uniaxial case.

components, so that

$$d\epsilon = d\epsilon_e + d\epsilon_p, \tag{2.34}$$

we define a strain-hardening parameter, H', as

$$H' = \frac{d\sigma}{d\epsilon_p}. \tag{2.35}$$

This can be interpreted as the slope of the strain-hardening portion of the stress–strain curve after removal of the elastic strain component. Thus

$$H' = \frac{d\sigma}{d\epsilon - d\epsilon_e} = \frac{E_T}{1 - E_T/E}. \tag{2.36}$$

With reference to Fig. 2.8, consider the behaviour of a linear displacement element, which has a cross-sectional area A, when it is subjected to a gradually increasing axial force, F, which results in an extension, δ. Provided that F/A is less than or equal to the uniaxial yield stress, σ_Y, the material behaviour will be elastic, exhibiting a stiffness of

$$K_e = \frac{F}{\delta} = \frac{EA}{L}, \tag{2.37}$$

then the element stiffness matrix is simply

$$K_e^{(e)} = \frac{EA}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{2.38}$$

Suppose F is increased until the material has yielded. Consider a further incremental increase in load dF which causes an additional element extension, $d\delta$. Then

$$d\delta = (d\epsilon_e + d\epsilon_p)L, \tag{2.39}$$

where L is the element length. Also, on use of (2.35)

$$dF = d\sigma A = AH' d\epsilon_p. \tag{2.40}$$

The tangential stiffness for the material is then

$$K_{ep} = \frac{dF}{d\delta} = \frac{AH'\, d\epsilon_p}{L(d\sigma/E + d\epsilon_p)}. \tag{2.41}$$

Or, using (2.35) and rearranging

$$K_{ep} = \frac{EA}{L}\left(1 - \frac{E}{E+H'}\right). \tag{2.42}$$

Finally, the element stiffness for elasto-plastic material behaviour is given by*

$$K_{ep}^{(e)} = \frac{EA}{L}\left(1 - \frac{E}{E+H'}\right)\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{2.43}$$

In (2.42) it can be seen that the first term represents the elastic stiffness, as given by (2.38). The second term accounts for the reduction in stiffness from the elastic value due to yielding.

* The element stiffness matrix can be written in the standard finite element form

$$K_e^{(e)} = \int_V \boldsymbol{B}^T \boldsymbol{D}\boldsymbol{B}\, dV = A\int_0^L \boldsymbol{B}^T \boldsymbol{D}\boldsymbol{B}\, dx,$$

where integration is made over the volume of the element. For this one-dimensional application, $\boldsymbol{D} = E$ and

$$\boldsymbol{B} = \left[\frac{dN_1^{(e)}}{dx}, \quad \frac{dN_2^{(e)}}{dx}\right] = \left[-\frac{1}{L}, \quad \frac{1}{L}\right],$$

where $N_1^{(e)}$ and $N_2^{(e)}$ are given by (2.24). The tangential stiffness matrix for elasto-plastic material behaviour is obtained by replacing $\boldsymbol{D}$ by

$$\boldsymbol{D}_{ep} = E\left(1 - \frac{E}{E+H'}\right).$$

For a perfectly plastic material behaviour, after initial yielding equation (2.36) implies that $H' = 0$ and it is then evident from (2.43) that $K_{ep}^{(e)} = 0$. This implies that the tangential (elasto-plastic) stiffness matrix for such a material is singular and the tangential stiffness method cannot generally be employed in solution. If a significant number of elements in the structure has yielded, the assembled tangential stiffness matrix will be singular, and the inversion or reduction demanded by (2.13) cannot be performed. This difficulty can be avoided by use of the initial stiffness method in which the elastic element stiffnesses are employed at every stage of the computation, thereby ensuring a positive definite assembled stiffness matrix.

2.6 Problems

In this section some tasks are set for the reader which illustrate some further points in connection with the topics discussed in the chapter.

2.1 Use the direct iteration method to solve the following one degree of freedom problem, $H\phi + f = 0$ where $f = 10$ and H depends on ϕ according to $H = 10(1+e^{3\phi})$.

2.2 Repeat Problem 2.1 using the Newton–Raphson method. Compare the solutions and the computational effort required in each.

2.3 Solve the following one degree of freedom problem by both the tangential stiffness and initial stiffness method. Apply the total load f as two equal increments

$$H\phi + f = 0, \quad f = 10, \quad H = 20(1-\phi).$$

2.4 The more general form of the boundary condition (2.17) in Section 2.3 is $d\phi/dx + q + \alpha.\phi = 0$, where q and α are constants and ϕ is the undetermined value of the unknown at the boundary point. Repeat the Galerkin process of Section 2.3 to include these additional terms. In particular, determine the additional nodal force contribution and the discrete 'external' nodal stiffness which arise.

2.5 For the two-noded element with linear variation in ϕ with shape functions as given by (2.24), evaluate the element stiffness matrix when K is a function of x. Assume that the spatial variation of K within the element is linear and obtained by interpolation of the specified nodal values by use of the element shape functions.

2.6 Suppose that a heat loss also occurs by convection from the surface area of an element, which is given by $h.\phi$ where h is the convection coefficient. If C is the circumference of the element, determine the additional contribution to $\boldsymbol{H}^{(e)}$ resulting from this.[9]

2.7 Determine the nonlinear portion, $\boldsymbol{H}'^{(e)}$, of the Jacobian matrix for a material dependence $K = K_0(1+e^{b\phi})$. Assume a two-noded linear element.

2.8 Evaluate the stiffness matrix $\boldsymbol{H}^{(e)}$ for a three-noded element for a heat conduction problem. Assume that the element has shape functions

$$N_1^{(e)} = -\frac{2x}{L^2}\left(\frac{L}{2}-x\right), \quad N_2^{(e)} = \frac{4}{L^2}\left(\frac{L}{2}-x\right)\left(\frac{L}{2}+x\right),$$

$$N_3^{(e)} = \frac{2x}{L^2}\left(\frac{L}{2}+x\right),$$

and also that $K = K_0(a+b\phi)$ where K_0, a and b are constants.

2.9 Repeat Problem 2.8 for the case where K_0 is additionally a function of x. Assume that the nodal values of K_0 are given.

2.10 Solve the nonlinear elastic problem of Fig. 2.10 by hand calculation. Use the tangential stiffness method and assume the total load to be applied in two equal increments.

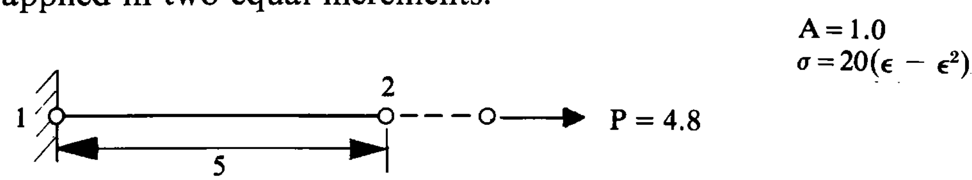


Fig. 2.10 Nonlinear elastic example—Problem 2.10.

2.11 Solve Problem 2.10 if the structure is loaded by incrementally increasing the prescribed value of displacement at node 2. Increase the applied displacement in two equal increments up to a maximum value of $\phi_2 = 3.0$. Since the element stiffnesses become negative at the higher increment, use the initial stiffness method.

2.12 A locking material is one in which the stiffness increases with increasing strains. For example, if $g(\epsilon) = \epsilon^2$ can both the tangential stiffness and the initial stiffness methods be used to solve such material problems?

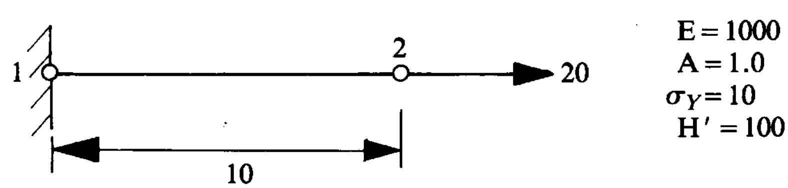
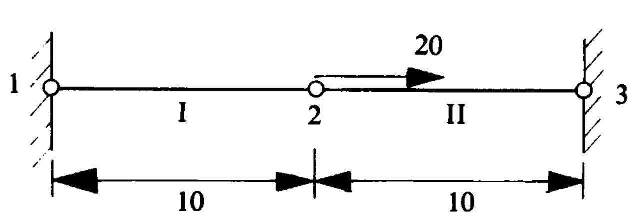


Fig. 2.11 Elasto-plastic example—Problem 2.13.

2.13 Determine the nodal displacement of node 2 of the structure shown in Fig. 2.11 as the applied load is increased to 10 units in two equal increments. Assume elasto-plastic material behaviour and use the tangential stiffness approach for solution.



	Element I	Element II
E	1000	1000
A	1.0	1.0
σ_Y	5.0	5.0
H'	200	−100

Fig. 2.12 Bimaterial elasto-plastic example—Problem 2.14.

2.14 Determine the displacement of node 2 of the elasto-plastic structure shown in Fig. 2.12. Assume the load to be applied in two equal increments. What happens if $H_I' = 200$, $H_{II}' = -200$?

2.7 References

1. ZIENKIEWICZ, O. C., *The Finite Element Method*, McGraw-Hill, London, 1977.
2. BOOTH, A. D., *Numerical Methods*, Butterworth, London, 1966.
3. ZIENKIEWICZ, O. C., VALLIAPPAN, S. and KING, I. P., Elasto-plastic solutions of engineering problems. Initial stress, finite element approach, *Int. J. Num. Meth. Engng.*, **1**, 75–100 (1969).
4. ARGYRIS, J. H. and SCHARPF, D. W., Methods of elasto-plastic analysis, *ISD, ISSC Symp. on Finite Element Tech.*, Stuttgart (1969).
5. LYNESS, J. F., OWEN, D. R. J. and ZIENKIEWICZ, O. C., The finite element analysis of engineering systems governed by a nonlinear quasi-harmonic equation, *Computers and Structures*, **5**, 65–79 (1975).
6. HINTON, E. and OWEN, D. R. J., *An Introduction to Finite Element Computations*, Pineridge Press, Swansea, U.K., 1979.
7. ODEN, J. T., *Finite Elements of Nonlinear Continua*, McGraw-Hill, New York, 1972.
8. HILL, R., *The Mathematical Theory of Plasticity*, Oxford University Press, 1950.
9. SEEGERLIND, L. J., *Applied Finite Element Analysis*, John Wiley, New York, 1976.

Chapter 3
Solution of nonlinear problems

3.1 Introduction

A modular approach is adopted for the programs presented in this text, with the various main finite element operations being performed by separate subroutines. Any nonlinear finite element program must essentially contain all the subroutines necessary for elastic analysis. Briefly these consist of a subroutine to accept the input data, a subroutine for element stiffness formulation, subroutines for equation assembly and solution and a subroutine for output of the final results.

In order to implement the solution algorithms described in Section 2.2, additional subroutines are clearly necessary. In particular two primary DO LOOPS are necessary to iterate the solution until convergence of the solution occurs and to increment the applied loading, if appropriate. Subroutines must be included to evaluate the residual forces and also to monitor convergence of the solution. Figure 3.1 shows the organisation of the programs presented in this chapter, particularly the sequence in which the subroutines are accessed. Four separate programs are developed to solve the following specific situations.

- Solution of nonlinear quasi-harmonic situations by direct iteration.
- Solution of nonlinear quasi-harmonic situations by the Newton–Raphson method.
- Solution of nonlinear elastic problems by either the tangential stiffness or the initial stiffness method or a combination of both.
- Solution of elasto-plastic problems by either the tangential stiffness or the initial stiffness method or a combination of both approaches.

With reference to Fig. 3.1, most of the subroutines are common to all four programs presented; the only exceptions being the subroutines necessary for stiffness matrix generation, residual force calculation and solution convergence checking. The element stiffness formulation subroutines for quasi-harmonic direct interation, quasi-harmonic Newton–Raphson, nonlinear elastic situations and elasto-plastic problems are respectively named STIFF1, ASTIF1, STIFF2 and STIFF3. The evaluation of residual forces is not required in the direct iteration method and the appropriate subroutines for the quasi-harmonic Newton–Raphson, nonlinear elastic and elasto-plastic

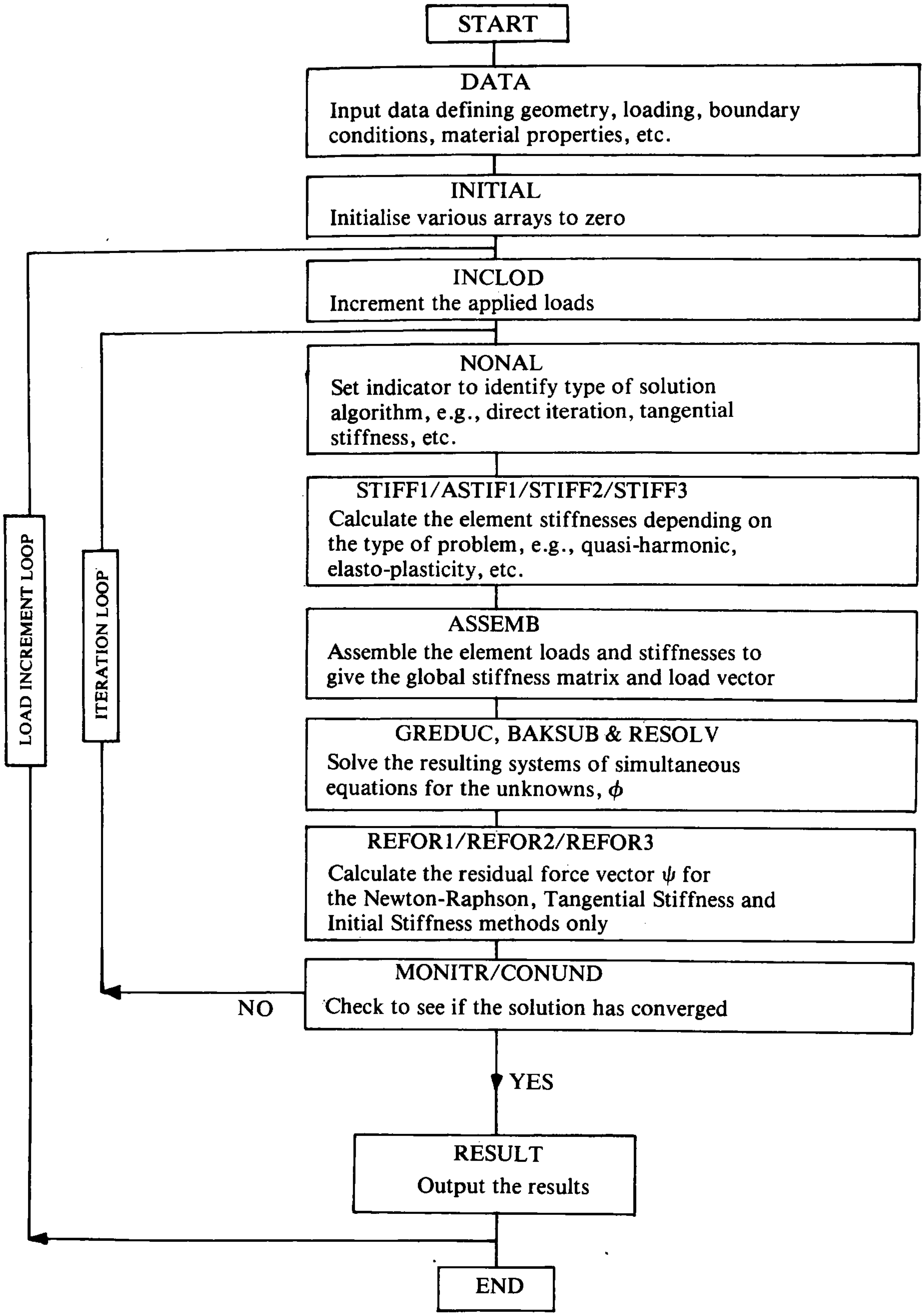


Fig. 3.1 Program organisation for one-dimensional nonlinear applications.

situations are named respectively REFOR1, REFOR2 and REFOR3. Finally, since the basis of solution convergence differs for the direct iteration method from that of the other procedures, it requires a separate convergence checking subroutine, termed MONITR. The equivalent subroutine for all other applications is named CONUND.

The programs presented in this chapter also form the basis of an elasto-viscoplastic program for one-dimensional applications developed in Chapter 4 and an elasto-plastic beam bending program considered in Chapter 5. In order to allow several of the subroutines developed in this chapter to be used for beam bending applications it will be necessary to permit the number of degrees of freedom per nodal point to be variable and to dimension some arrays to accommodate additional quantities.

Sections 3.2 to 3.8 are devoted to the development of the subroutines which are common to the four programs presented.

3.2 Input data subroutine, DATA

For any finite element analysis the input data can be subdivided into three main classifications. Firstly the data required to define the geometry of the structure and the support conditions must be supplied. Secondly the material properties of the constituent materials must be supplied and finally the applied loading must be furnished.

To allow a subroutine to be employed in more than one application, several control parameters must be supplied as input data. For example, the number of properties required to define the behaviour of a material will differ between quasi-harmonic problems and elasto-plastic situations. The use of variables in place of specific numerical values also generally aids program clarity.

A list of control parameters required as input is now presented:

NPOIN Total number of nodal points in the structure.

NELEM Total number of elements in the structure.

NBOUN Total number of boundary points, i.e. nodal points at which the value of the unknown is prescribed. In this context an internal node can be a boundary node.

NMATS Total number of different materials in the structure.

NPROP The number of material parameters required to define the characteristics of a material completely:
4—For elasto-plastic problems,
2—For all other applications.

NNODE Number of nodes per element. For linear displacement one-dimensional elements this equals 2.

NINCS The number of increments in which the total loading is to be applied.

NALGO Indicator used to identify the type of solution algorithm to be employed:
1—Direct iteration.

2—Newton–Raphson method for quasi-harmonic problems. Tangential stiffness method for structural problems (nonlinear elastic and elasto-plastic situations).
3—Initial stiffness method.
4—Combination of the initial and tangential stiffness methods, where the stiffnesses are recalculated on the first iteration of a load increment only.
5—Combination of the initial and tangential stiffness methods, where the stiffnesses are recalculated on the second iteration of a load increment only. This can aid the rate of convergence considerably, if on the application of an increment of load there is substantial further yielding. When calculating the element stiffnesses the total plastic strains evaluated during the previous iteration are used to indicate whether the element has yielded or not. If the element stiffnesses are recalculated on the first iteration, the elements which have now yielded may have been elastic at the end of the previous load increment and consequently the reformulated stiffness will be based on elastic behaviour. This can reduce the convergence rate of the process since generally $H' \simeq 0.1E$. From (2.42) the elasto-plastic stiffness is proportional to $E(1-E/(E+H')) \simeq E/11$, whereas the elastic stiffness depends linearly on E. Hence the tangential stiffness calculated grossly overestimates the true material response. This problem can be alleviated by reformulating the element stiffnesses during the second iteration of a load increment rather than the first, since the plastic strain evaluated on the first iteration will indicate yielding to have initiated.

NDOFN The number of degrees of freedom per nodal point:
1—For uniaxial problems.
2—For beam bending problems (considered in Chapter 5).

The geometry of the structure is completely defined on prescription of the nodal point coordinates and the element nodal connections. The coordinate of each nodal point must be defined with reference to a global coordinate system. For the one-dimensional situation being currently considered, the position of each nodal point is completely defined by a single coordinate whose value will be stored in the array

COORD (IPOIN)

where IPOIN corresponds to the number of the nodal point.

The origin of the coordinate system can be arbitrarily chosen. The geometry of each individual element must be specified by listing in a systematic way the numbers of the nodal points which define its outline. For the two-noded linear displacement element the nodal numbers can obviously be read in any order. The element topology is read into the array

LNODS (NUMEL, INODE)

where NUMEL corresponds to the number of the element under consideration and subscript INODE ranges from 1 to NNODE. Since each element may conceivably be assigned different material properties, a material property identification number is also allocated to each element and stored in the array

MATNO (NUMEL)

This implies that element number NUMEL has material properties of type MATNO (NUMEL).

The material properties required for solution will differ for the various applications considered, but the same array will be employed for storage of this information. Namely

PROPS (NUMAT, IPROP)

where NUMAT denotes the material identification number and the subscript IPROP the individual property. Each element is associated with a particular material type through the previously mentioned identification array MATNO (NUMEL). The relevant material properties associated with the different problem types considered here are listed below.

(a) Quasi-harmonic problems

PROPS (NUMAT, 1)—The reference value K_0 of the coefficient K in equation (2.27).
PROPS (NUMAT, 2)—The constant b in equation (2.27) for a linear 'stiffness' variation.

(b) Nonlinear elastic problems

PROPS (NUMAT, 1)—The reference value E_0 in (2.30).
PROPS (NUMAT, 2)—The cross-sectional area A, of the element. Each element with a different cross-sectional area must be assigned a different material property number.

(c) Elasto-plastic problems

PROPS (NUMAT, 1)—The elastic modulus, E, of the material.
PROPS (NUMAT, 2)—The cross-sectional area, A, of the element.
PROPS (NUMAT, 3)—The uniaxial yield stress of the material.
PROPS (NUMAT, 4)—The linear strain hardening parameter, H', for the material (equation (2.35)).

It should be mentioned here that the specific form of dependence of material stiffness on the unknown function for cases (a) and (b) will be directly incorporated into the program by use of a FORTRAN FUNCTION statement.

Any nodal points at which a degree of freedom has a prescribed value must be identified by the temporary variable NODFX. To determine which degrees of freedom are to be prescribed at this node, the entries in the array

ICODE (IDOFN)

are set to either 0 or 1. (Variable IDOFN ranges over the number of degrees of freedom per node NDOFN. In the present case NDOFN = 1, but later in Chapter 5, NDOFN has the value 2.) If ICODE (IDOFN) is equal to 1, then degree of freedom IDOFN at node NODFX has a prescribed value. If NCODE (IDOFN) is equal to 0 then degree of freedom IDOFN at node NODFX is a free variable.

The value for a prescribed degree of freedom is given by

VALUE (IDOFN)

It should be noted that if ICODE (IDOFN) = 0, then VALUE (IDOFN) is ignored.

In order to simplify the solution process, the information stored in arrays ICODE and VALUE is transferred to much larger arrays IFPRE (NPOSN) and PEFIX (NPOSN) respectively, where NPOSN ranges over all the degrees of freedom for the whole finite element mesh. Both IFPRE and PEFIX are initially set equal to zero and as data for each restrained boundary node is read, they are modified if necessary. Unit entries in IFPRE indicate that the associated variable is prescribed. The prescribed value is obtained from the corresponding position in PEFIX.

Finally, the loads applied to the structure must be specified. For the *frontal method of equation solution* employed in later chapters it is convenient to associate the applied loads with the elements on which they act. Thus for each element the nodal loads acting on the two nodes associated with the element must be input and these are stored in the array

RLOAD (IELEM, IEVAB)

where IELEM indicates the element number and IEVAB relates to the degrees of freedom of the element (IEVAB ranges from 1 to NEVAB, the number of element variables, which is equal to 2 in the present case but which equals 4 in the applications described in Chapter 5). It should be noted that a nodal load may be arbitrarily assigned to any one of the elements connected to that node, since before eventual solution all element contributions are assembled to form a global load vector. Before entering the solution routines the loads are transferred to an array ELOAD (IELEM, IEVAB) as described later in Section 3.7.

Subroutine DATA is now presented and should be largely self-explanatory. Descriptive comments are provided immediately after the FORTRAN listing of the subroutine.

```
      SUBROUTINE  DATA
C***********************************************************************
C
C *** INPUTS DATA DEFINING GEOMETRY,LOADING,BOUNDARY CONDITIONS...ETC.
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      DIMENSION ICODE(2),VALUE(2),TITLE(18)
      READ (5,965)TITLE
      WRITE(6,965)TITLE
  965 FORMAT(18A4)
      READ(5,900) NPOIN,NELEM,NBOUN,NMATS,NPROP,NNODE,NINCS,NALGO,NDOFN
  900 FORMAT(9I5)
      WRITE(6,905)NPOIN,NELEM,NBOUN,NMATS,NPROP,NNODE,NINCS,NALGO,NDOFN
  905 FORMAT(//1X,'NPOIN =',I5,3X,'NELEM =',I5,3X,'NBOUN =',I5,3X,
     .            'NMATS =',I5//1X,'NPROP =',I5,3X,'NNODE =',I5,3X,
     .            'NINCS =',I5,3X,'NALGO =',I5//1X,'NDOFN =',I5)
      NEVAB=NDOFN*NNODE
      NSVAB=NDOFN*NPOIN
      WRITE(6,910)
  910 FORMAT(1H0,5X,'MATERIAL PROPERTIES')
      DO 10 IMATS=1,NMATS
      READ (5,915) JMATS,(PROPS(JMATS,IPROP),IPROP=1,NPROP)
   10 WRITE(6,915) JMATS,(PROPS(JMATS,IPROP),IPROP=1,NPROP)
  915 FORMAT(I10,4F15.5)
      WRITE(6,920)
  920 FORMAT(1H0,3X,'EL   NODES   MAT.')
      DO 20 IELEM=1,NELEM
      READ (5,925) JELEM,(LNODS(JELEM,INODE),INODE=1,NNODE),MATNO(JELEM)
   20 WRITE(6,925) JELEM,(LNODS(JELEM,INODE),INODE=1,NNODE),MATNO(JELEM)
  925 FORMAT(4I5)
      WRITE(6,930)
  930 FORMAT(1H0,5X,'NODE',5X,'COORD.')
      DO 30 IPOIN=1,NPOIN
      READ (5,935) JPOIN,COORD(JPOIN)
   30 WRITE(6,935) JPOIN,COORD(JPOIN)
  935 FORMAT(I10,F15.5)
      DO 40 ISVAB=1,NSVAB
      IFPRE(ISVAB)=0
   40 PEFIX(ISVAB)=0.0
      IF(NDOFN.EQ.1) WRITE(6,940)
  940 FORMAT(1H0,1X,'RES.NODE',2X,'CODE',3X,'PRES.VALUES')
      IF(NDOFN.EQ.2) WRITE(6,945)
  945 FORMAT(1H0,1X,'RES.NODE',2X,'CODE',3X,'PRES.VALUES',2X,
     .       'CODE',3X,'PRES.VALUES')
      DO 50 IBOUN=1,NBOUN
      READ (5,950) NODFX,(ICODE(IDOFN),VALUE(IDOFN),IDOFN=1,NDOFN)
      WRITE(6,950) NODFX,(ICODE(IDOFN),VALUE(IDOFN),IDOFN=1,NDOFN)
  950 FORMAT(I10,2(I5,F15.5))
      NPOSN=(NODFX-1)*NDOFN
      DO 50 IDOFN=1,NDOFN
      NPOSN=NPOSN+1
      IFPRE(NPOSN)=ICODE(IDOFN)
   50 PEFIX(NPOSN)=VALUE(IDOFN)
      WRITE(6,955)
  955 FORMAT(1H0,2X,'ELEMENT',10X,'NODAL LOADS')
      DO 60 IELEM=1,NELEM
```

```
      DO 60 IEVAB=1,NEVAB                                      DATA 65
   60 RLOAD(IELEM,IEVAB)=0.0                                   DATA 66
   70 READ (5,960) JELEM,(RLOAD(JELEM,IEVAB),IEVAB=1,NEVAB)    DATA 67
      IF(JELEM.NE.NELEM) GO TO 70                              DATA 68
      DO 80 IELEM=1,NELEM                                      DATA 69
   80 WRITE(6,960) IELEM,(RLOAD(IELEM,IEVAB),IEVAB=1,NEVAB)    DATA 70
  960 FORMAT(I10,5F15.5)                                       DATA 71
      RETURN                                                   DATA 72
      END                                                      DATA 73
```

DATA 16–18 Read and write the problem title.

DATA 19–24 Read and write the control parameters for the problem.

DATA 27–32 Read and write the material properties for each individual material.

DATA 33–38 Read and write the nodal connection numbers and material identification number of each element.

DATA 39–47 Read and write the coordinate of each nodal point. Also initialise the arrays for locating and recording prescribed values of the unknown.

DATA 48–61 Read and write the node number and prescribed value for each degree of freedom for each boundary node and store in the global arrays IFPRE and PEFIX.

DATA 62–71 Read and write the nodal loads for each element.

3.3 Subroutine NONAL

The main function of this subroutine is to control the solution process according to the value of the solution algorithm parameter, NALGO, input in subroutine DATA. The subroutine sets the value of indicator KRESL to either 1 or 2 according to NALGO and the current value of the iteration number IITER and increment number IINCS. A value of KRESL = 1 indicates that the stiffnesses are to be reformulated and consequently a full system of simultaneous equations must be subsequently solved. If KRESL = 2 the stiffnesses are not to be redefined and therefore only equation resolution need be undertaken. In this the reduced equations from the previous solution are stored and only the terms associated with the new loading need be reduced in the solution process. This results in a considerable saving in computation time with equation resolution generally requiring only 20% of the time required for complete analysis. For the algorithm options contained in the four programs presented, the value of KRESL is preset as follows.

(a) *Direct iteration.* For this case the stiffnesses must be reformulated, according to (2.3), for every iteration. Consequently KRESL = 1 at all stages.

(b) *Newton–Raphson method for quasi-harmonic problems and tangential stiffness method for structural problems.* Again the stiffnesses must be reformulated for every iteration according to (2.12) for quasi-harmonic situations and (2.13) for structural applications. Therefore KRESL = 1 at all stages.

(c) *Initial stiffness method.* In this approach the stiffnesses are calculated once and for all at the beginning of the computation, according to (2.14) and this value is then used throughout. Consequently KRESL = 1 for the first iteration of the first load increment and is set equal to 2 thereafter.

(d) *Combination of initial and tangential stiffness methods.* In this algorithm the stiffnesses are recalculated only for the first iteration of any load increment and kept constant thereafter until convergence of solution under that particular loading is achieved. Therefore KRESL = 1 for the first iteration of any load increment and is set to 2 at all other times. (Alternatively the element stiffnesses may be recomputed at the beginning of the *second* iteration as described in Section 3.2.)

The final role of subroutine NONAL is to set the vector of prescribed unknowns to the correct values. For the method of direct iteration the problem is completely reanalysed for every iteration and therefore the vector of prescribed unknowns must be introduced unchanged into the solution subroutines at each stage. However, for the three other solution algorithms considered, the processes are essentially accumulative with the value of the unknowns being totalled from the incremental values obtained for each iteration. Therefore, in order to maintain the fixed unknowns at their prescribed values, it is necessary to input the prescribed values into the solution routines for the first iteration of a load increment and then prescribe zero values for all subsequent iterations. In this way the final displacements will equal the prescribed values on convergence of the solution. If the structure is to be loaded by prescribing values of the unknowns then an incremental procedure may be adopted with factored values of the prescribed unknowns being applied sequentially. The prescribed displacements are factored by use of the variable FACTO, whose role is explained in terms of applied loads in Section 3.7. The prescribed values of the unknowns have been permanently stored in array PEFIX in subroutine DATA. These prescribed values, or zero values, required as described above, are transferred to the equation solution subroutines via the array FIXED.

Subroutine NONAL is now presented and explanatory notes provided.

```
      SUBROUTINE  NONAL
C***********************************************************************
C
C *** SETS INDICATOR TO IDENTIFY TYPE OF SOLUTION ALGORITHM
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
```

```
     .         TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),          NONL  13
     .         REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)               NONL  14
      KRESL=2                                                           NONL  15
      IF(NALGO.EQ.1) KRESL=1                                            NONL  16
      IF(NALGO.EQ.2) KRESL=1                                            NONL  17
      IF(NALGO.EQ.3.AND.IINCS.EQ.1.AND.IITER.EQ.1) KRESL=1              NONL  18
      IF(NALGO.EQ.4.AND.IITER.EQ.1) KRESL=1                             NONL  19
      IF(NALGO.EQ.5.AND.IINCS.EQ.1.AND.IITER.EQ.1) KRESL=1              NONL  20
      IF(NALGO.EQ.5.AND.IITER.EQ.2) KRESL=1                             NONL  21
      IF(IITER.EQ.1.OR.NALGO.EQ.1) GO TO 20                             NONL  22
      DO 10 ISVAB=1,NSVAB                                               NONL  23
   10 FIXED(ISVAB)=0.0                                                  NONL  24
      RETURN                                                            NONL  25
   20 DO 30 ISVAB=1,NSVAB                                               NONL  26
   30 FIXED(ISVAB)=PEFIX(ISVAB)*FACTO                                   NONL  27
      RETURN                                                            NONL  28
      END                                                               NONL  29
```

NONL 15	Preset KRESL to the condition of equation resolution.
NONL 16	For the *direct iteration method* set KRESL = 1 for recomputation of the stiffnesses at all stages.
NONL 17	For the *Newton–Raphson method* for quasi-harmonic problems or the *tangential stiffness method* for structural problems, recompute the stiffnesses at all stages.
NONL 18	For the *initial stiffness method* for structural problems, compute the stiffnesses only at the beginning of the computation procedure.
NONL 19	For the *combined initial and tangential stiffness approach* and NALGO = 4, recompute the stiffnesses at the first iteration of each load increment only.
NONL 20–21	For the initial/tangential approach with the option NALGO = 5 (Section 3.2), the stiffnesses are recalculated on the 2nd iteration of any load increment. However, at the start of the computation the stiffnesses must be evaluated.
NONL 22	For all stages of the direct iteration method or the first iteration of the other techniques, go to 20 to set the unknowns equal to the prescribed values.
NONL 23–25	Set the vector of prescribed unknowns to zero and return.
NONL 26–27	Set the vector of prescribed unknowns equal to the input prescribed values multiplied by a specified factor.

3.4 Subroutines for equation assembly and solution

For finite element analysis by the displacement process, the stiffness and load contributions of each element must be assembled into the global stiffness matrix and load vector respectively. The resulting set of simultaneous equations must then be solved to give the unknown nodal values. These aspects have been dealt with in detail elsewhere[1–3] and only the essential steps of the process will be reproduced here.

3.4.1 Numerical example of equation assembly and solution

In order to introduce the global stiffness matrix assembly and equation solution process we consider the example of a simple axial load structure shown in Fig. 3.2. The structure is subdivided into four elements in each of which a linear displacement variation is assumed. At each node i of the element there is an axial displacement degree of freedom, ϕ_i.

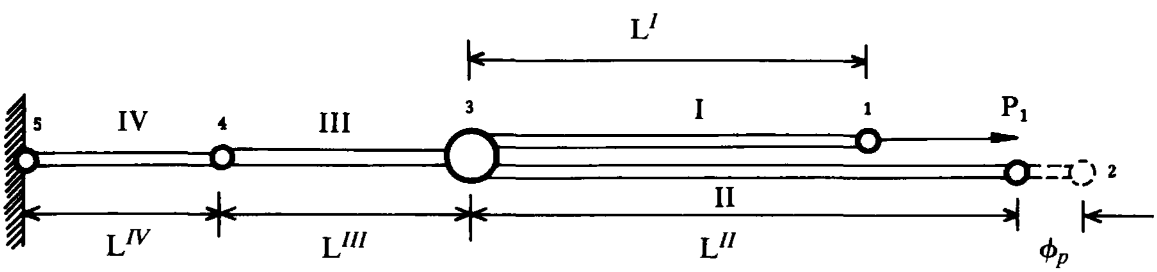


Fig. 3.2 Structural example for illustration of equation solution process.

The stiffness matrix for this element has already been derived in Section 2.5 and is given, for elastic material behaviour, by equation (2.38). The element stiffness matrices can be written as

$$\boldsymbol{K}_{\mathrm{I}} = k_{\mathrm{I}}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad \boldsymbol{K}_{\mathrm{II}} = k_{\mathrm{II}}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix},$$

$$\boldsymbol{K}_{\mathrm{III}} = k_{\mathrm{III}}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \quad \boldsymbol{K}_{\mathrm{IV}} = k_{\mathrm{IV}}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \tag{3.1}$$

where

$$k_{\mathrm{I}} = \frac{E^{(\mathrm{I})}A^{(\mathrm{I})}}{L^{(\mathrm{I})}}, \text{ etc.,} \tag{3.2}$$

in which $E^{(\mathrm{I})}$, $A^{(\mathrm{I})}$ and $L^{(\mathrm{I})}$ are respectively the elastic modulus, cross-sectional area and length of element I. The vector of applied nodal forces for each element is

$$\boldsymbol{f}_{\mathrm{I}} = \begin{bmatrix} P_1 \\ 0 \end{bmatrix}, \quad \boldsymbol{f}_{\mathrm{II}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{f}_{\mathrm{III}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{f}_{\mathrm{IV}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{3.3}$$

The vectors of the unknown nodal displacements for the elements are

$$\boldsymbol{\delta}_{\mathrm{I}} = \begin{bmatrix} \phi_1 \\ \phi_3 \end{bmatrix}, \quad \boldsymbol{\delta}_{\mathrm{II}} = \begin{bmatrix} \phi_2 \\ \phi_3 \end{bmatrix}, \quad \boldsymbol{\delta}_{\mathrm{III}} = \begin{bmatrix} \phi_3 \\ \phi_4 \end{bmatrix}, \quad \boldsymbol{\delta}_{\mathrm{IV}} = \begin{bmatrix} \phi_4 \\ \phi_5 \end{bmatrix}. \tag{3.4}$$

We also assume the following prescribed displacement values

$$\phi_2 = \phi_p, \quad \phi_5 = 0. \tag{3.5}$$

The Theorem of Minimum Total Potential Energy will now be used to derive the stiffness equations for this problem. The total potential energy for each element may be calculated separately. For example, the total potential energy of element I can be expressed as

$$\pi_{\mathrm{I}} = \tfrac{1}{2}[\delta_{\mathrm{I}}]^T K_{\mathrm{I}}\delta_{\mathrm{I}} - [\delta_{\mathrm{I}}]^T f_{\mathrm{I}} = \frac{k_{\mathrm{I}}}{2}(\phi_1-\phi_3)^2 - P_1\phi_1. \tag{3.6}$$

The augmented total potential energy of the assemblage is given by the sum of the individual element potentials plus extra terms to account for the prescribed values

$$\pi = \pi_{\mathrm{I}}+\pi_{\mathrm{II}}+\pi_{\mathrm{III}}+\pi_{\mathrm{IV}} - R_2(\phi_2-\phi_p) - R_5(\phi_5-0) \tag{3.7}$$

Note that R_2 and R_5 are the associated nodal reactions.

Using the principle of minimum potential energy, we obtain

$$\begin{aligned}
\frac{\partial\pi}{\partial\phi_1} &= k_{\mathrm{I}}(\phi_1-\phi_3) - P_1 = 0,\\
\frac{\partial\pi}{\partial\phi_2} &= k_{\mathrm{II}}(\phi_2-\phi_3) = R_2,\\
\frac{\partial\pi}{\partial\phi_3} &= k_{\mathrm{I}}(\phi_3-\phi_1)+k_{\mathrm{II}}(\phi_3-\phi_2)+k_{\mathrm{III}}(\phi_3-\phi_4) = 0,\\
\frac{\partial\pi}{\partial\phi_4} &= k_{\mathrm{III}}(\phi_4-\phi_3)+k_{\mathrm{IV}}(\phi_4-\phi_5) = 0,\\
\frac{\partial\pi}{\partial\phi_5} &= k_{\mathrm{IV}}(\phi_5-\phi_4) = R_5.
\end{aligned} \tag{3.8}$$

These equilibrium equations for the assembled elements of the structure can be expressed in matrix form as

$$\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 1\\2\\3\\4\\5 \end{array} &
\begin{bmatrix}
k_{\mathrm{I}} & 0 & -k_{\mathrm{I}} & 0 & 0\\
0 & k_{\mathrm{II}} & -k_{\mathrm{II}} & 0 & 0\\
-k_{\mathrm{I}} & -k_{\mathrm{II}} & k_{\mathrm{I}}+k_{\mathrm{II}}+k_{\mathrm{III}} & -k_{\mathrm{III}} & 0\\
0 & 0 & -k_{\mathrm{III}} & k_{\mathrm{III}}+k_{\mathrm{IV}} & -k_{\mathrm{IV}}\\
0 & 0 & 0 & -k_{\mathrm{IV}} & k_{\mathrm{IV}}
\end{bmatrix}
\end{array}
\begin{bmatrix}\phi_1\\ \phi_2\\ \phi_3\\ \phi_4\\ \phi_5\end{bmatrix}
=
\begin{bmatrix}P_1\\ R_2\\ 0\\ 0\\ R_5\end{bmatrix} \tag{3.9}$$

The assembly process can be clearly appreciated by comparing the individual stiffness matrices (3.1), and load vectors (3.3), with the final assemblage. Obviously, the individual element contributions can be added directly into the overall stiffness matrix of the structure in positions appropriate to the element nodal connection numbers.

It is noted that the global stiffness matrix is both symmetric and banded. By banded we mean that all the non-zero stiffness coefficients lie within a band adjacent to the leading diagonal. Banding of the stiffness equations is a direct consequence of the order in which the nodal points are numbered.

In the equation solution subroutines presented later in Sections 3.4.2–3.4.5 no advantage will be taken of the banded symmetric form of the stiffness equations.

Some elementary concepts of equation solution are now introduced. In particular we describe the Gaussian direct elimination process which will be used in a more efficient form in the main solution routine described later in Chapter 6.

3.4.1.1 Gaussian direct elimination method for the solution of simultaneous equation systems

Formulation of the global stiffness matrix resulted in equation system (3.9) which is of the general form

$$\begin{aligned}
k_{11}\phi_1+k_{12}\phi_2+k_{13}\phi_3+\ldots k_{1n}\phi_n &= f_1\\
k_{21}\phi_1+k_{22}\phi_2+k_{23}\phi_3+\ldots k_{2n}\phi_n &= f_2\\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
k_{n1}\phi_1+k_{n2}\phi_2+k_{n3}\phi_3+\ldots k_{nn}\phi_n &= f_n.
\end{aligned} \tag{3.10}$$

The Gaussian direct elimination method seeks to reduce equation system (3.10) to the following triangular form[4]

$$\begin{aligned}
k_{11}'\phi_1+k_{12}'\phi_2+k_{13}'\phi_3+\ldots k'_{1,n-1}\phi_{n-1}+k'_{1n}\phi_n &= f_1'\\
0\quad +k_{22}'\phi_2+k_{23}'\phi_3+\ldots k'_{2,n-1}\phi_{n-1}+k'_{2n}\phi_n &= f_2'\\
0\quad +\quad 0\quad +k_{33}'\phi_3+\ldots k'_{3,n-1}\phi_{n-1}+k'_{3n}\phi_n &= f_3'\\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
k'_{n-1,n-1}\phi_{n-1}+k'_{n-1,n}\phi_n &= f'_{n-1}\\
k'_{nn}\phi_n &= f_n'.
\end{aligned} \tag{3.11}$$

Then all the unknowns can be systematically determined by taking these *reduced* equations in reverse order, since each new equation, proceeding in an upward direction, only introduces one additional unknown value. The last equation is solved for ϕ_n, then ϕ_{n-1} can be recovered from the next equation and so on. This phase of the solution scheme is termed *back-substitution.*

3.4.1.2 The equation reduction or elimination phase

Reduction of system (3.10) to the form (3.11) can be accomplished by employing the i^{th} equation to eliminate ϕ_i from all equations below, i.e. from equations $i+1$ to n. Formally this can be done by subtracting from the r^{th} equation $(i < r \leqslant n)$, the i^{th} equation factored by $k_{ri}^{(i)}/k_{ii}^{(i)}$, where the superscript i indicates that these coefficients have been already modified $(i-1)$ times prior to the elimination of the i^{th} degree of freedom. For example, the first equation is used to eliminate ϕ_1 from equations 2 to n as follows:

$$
\begin{array}{l}
k_{11}\phi_1 + k_{12}\phi_2 + k_{13}\phi_3 + \ldots \quad k_{1n}\phi_n = f_1 \\
0.\phi_1 + \left(k_{22} - k_{12}\dfrac{k_{21}}{k_{11}}\right)\phi_2 + \left(k_{23} - k_{13}\dfrac{k_{21}}{k_{11}}\right)\phi_3 + \ldots \left(k_{2n} - k_{1n}\dfrac{k_{21}}{k_{11}}\right)\phi_n = f_2 - f_1\dfrac{k_{21}}{k_{11}} \\
\cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
0.\phi_1 + \left(k_{n2} - k_{12}\dfrac{k_{n1}}{k_{11}}\right)\phi_2 + \left(k_{n3} - k_{13}\dfrac{k_{n1}}{k_{11}}\right)\phi_3 + \ldots \left(k_{nn} - k_{1n}\dfrac{k_{n1}}{k_{11}}\right)\phi_n = f_n - f_1\dfrac{k_{n1}}{k_{11}}.
\end{array}
\tag{3.12}
$$

Then the second equation is used to eliminate ϕ_2 from equations 3 to n and so on. Note that the modified terms in the equation system are still symmetric.

3.4.1.3 The case of a prescribed displacement

If a displacement is prescribed its value is known. Therefore the nodal force necessary to maintain the specified displacement becomes the unknown value associated with the node. Suppose for example that ϕ_2 is prescribed to be some given value ϕ_p, in which case f_2 is the reaction value. In this case the elimination of ϕ_2 is trivial and all that need be done is to substitute $\phi_2 = \phi_p$ in equations 3 to n and transfer the now known quantity

$$k_{r2}'\phi_p \quad (3 \leqslant r \leqslant n)$$

to the right-hand side of each equation. This is illustrated below

$$
\begin{array}{l}
k_{11}\phi_1 + k_{12}\phi_2 + k_{13}\phi_3 + \ldots k_{1n}\phi_n = f_1 \\
0.\phi_1 + k_{22}'\phi_2 + k_{23}'\phi_3 + \ldots k_{2n}'\phi_n = f_2 \\
0.\phi_1 + 0.\phi_2 + k_{33}'\phi_3 + \ldots k_{3n}'\phi_n = f_3 - k_{32}'\phi_p \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\
0.\phi_1 + 0.\phi_2 + k_{n3}'\phi_3 + \ldots k_{nn}'\phi_n = f_n - k_{n2}'\phi_p.
\end{array}
\tag{3.13}
$$

For the particular case of a zero prescribed displacement value due to a pinned support, an alternative approach is to delete the row and column corresponding to the zero displacement from the equation system. The column can be deleted since it always multiplies a zero quantity and the row is removed since it only relates to equilibrium at the supported node. However this means that if the support reaction is required, it must be computed separately from the element forces meeting at the pinned node.

The complete solution process is best illustrated by application to a particular problem. We will now substitute explicit values for the terms contained in (3.9) in order to permit numerical solution. Assume that

$$k_{\text{I}} = 1, \quad k_{\text{II}} = 2, \quad k_{\text{III}} = 3, \quad k_{\text{IV}} = 4, \quad P_1 = 10, \quad \phi_p = 2, \tag{3.14}$$

then equations (3.9) can be written as

$$\phi_1+0.\phi_2-\ \ \phi_3+0.\phi_4+0.\phi_5 = 10 \tag{3.15a}$$

$$0.\phi_1+\ 2\phi_2-\ 2\phi_3+0.\phi_4+0.\phi_5 = R_2;\quad \phi_2 = 2 \tag{3.15b}$$

$$-\phi_1-2\phi_2\ +\ 6\phi_3-\ 3\phi_4+0.\phi_5 = 0 \tag{3.15c}$$

$$0.\phi_1+0.\phi_2-\ 3\phi_3+\ 7\phi_4-\ 4\phi_5 = 0 \tag{3.15d}$$

$$0.\phi_1+0.\phi_2+0.\phi_3-\ 4\phi_4+\ 4\phi_5 = R_5;\quad \phi_5 = 0. \tag{3.15e}$$

where R_2 and R_5 are the nodal reactions associated with the displacement values prescribed at nodes 2 and 5. For example, R_2 must balance the sum of the elastic forces provided by all the elements meeting at node 2. We also imply by the notation adopted that $\phi_2 = 2$.

To solve these equations by the Gaussian reduction process we first eliminate ϕ_1 from all equations, except (3.15a). Then we eliminate ϕ_2 from all equations below (3.15b), then ϕ_3 is eliminated from all equations below (3.15c) and so on. Therefore, we eliminate a particular variable only below the current or active equation. (If we are eliminating ϕ_r, the r^{th} equation is active.)

We commence the process by eliminating ϕ_1 from equations (3.15b)–(3.15e) by using (3.15a). In fact, we need only operate on (3.15c) since ϕ_1 does not appear in the other equations. Thus we eliminate ϕ_1 from (3.15c) by adding (3.15a) to (3.15c). This gives the first reduced set of equations as

$$\phi_1+0.\phi_2-\ \ \phi_3+0.\phi_4+0.\phi_5 = 10 \tag{3.16a}$$

$$0.\phi_1+\ 2\phi_2-\ 2\phi_3+0.\phi_4+0.\phi_5 = R_2;\quad \phi_2 = 2 \tag{3.16b}$$

$$0.\phi_1-\ 2\phi_2+\ 5\phi_3-\ 3\phi_4+0.\phi_5 = 10 \tag{3.16c}$$

$$0.\phi_1+0.\phi_2-\ 3\phi_3+\ 7\phi_4-\ 4\phi_5 = 0 \tag{3.16d}$$

$$0.\phi_1+0.\phi_2+0.\phi_3-\ 4\phi_4+\ 4\phi_5 = R_5;\quad \phi_5 = 0. \tag{3.16e}$$

Next we eliminate ϕ_2 from (3.16c)–(3.16e) by using (3.16b). In fact, since ϕ_2 is prescribed to be 2, all we need do is substitute $\phi_2 = 2$ directly into the remaining equations. We also do this for (3.16b) in this case.

$$\phi_1+0.\phi_2-\ \ \phi_3+0.\phi_4+0.\phi_5 = 10 \tag{3.17a}$$

$$0.\phi_1+0.\phi_2-\ 2\phi_3+0.\phi_4+0.\phi_5 = -4+R_2;\quad \phi_2 = 2 \tag{3.17b}$$

$$0.\phi_1+0.\phi_2+\ 5\phi_3-\ 3\phi_4+0.\phi_5 = 14 \tag{3.17c}$$

$$0.\phi_1+0.\phi_2-\ 3\phi_3+\ 7\phi_4-\ 4\phi_5 = 0 \tag{3.17d}$$

$$0.\phi_1+0.\phi_2+0.\phi_3-\ 4\phi_4+\ 4\phi_5 = R_5; \quad \phi_5 = 0. \tag{3.17e}$$

We then use (3.17c) to eliminate ϕ_3 from (3.17d) and (3.17e). We need only operate on (3.17d), since ϕ_3 does not appear in (3.17e), and in particular we add (3.17d) to 3/5 of (3.17c).

$$\phi_1+0.\phi_2-\ \phi_3+0.\phi_4+0.\phi_5 = 10 \tag{3.18a}$$

$$0.\phi_1+0.\phi_2-\ 2\phi_3+0.\phi_4+0.\phi_5 = -4+R_2; \quad \phi_2 = 2 \tag{3.18b}$$

$$0.\phi_1+0.\phi_2+\ 5\phi_3-\ 3\phi_4+\ \phi_5 = 14 \tag{3.18c}$$

$$0.\phi_1+0.\phi_2+0.\phi_3+\frac{26}{5}\phi_4-\ 4\phi_5 = \frac{42}{5} \tag{3.18d}$$

$$0.\phi_1+0.\phi_2+0.\phi_3-\ 4\phi_4+\ 4\phi_5 = R_5; \quad \phi_5 = 0. \tag{3.18e}$$

To complete the *elimination* process, we eliminate ϕ_4 from (3.18e) by adding (3.18e) to 20/26 of (3.18d).

$$\phi_1+0.\phi_2-\ \phi_3+0.\phi_4+0.\phi_5 = 10 \tag{3.19a}$$

$$0.\phi_1+0.\phi_2-\ 2\phi_3+0.\phi_4+0.\phi_5 = -4+R_2; \quad \phi_2 = 2 \tag{3.19b}$$

$$0.\phi_1+0.\phi_2+\ 5\phi_3-\ 3\phi_4+\ \phi_5 = 14 \tag{3.19c}$$

$$0.\phi_1+0.\phi_2+0.\phi_3+\frac{26}{5}\phi_4-\ 4\phi_5 = \frac{42}{5} \tag{3.19d}$$

$$0.\phi_1+0.\phi_2+0.\phi_3+0.\phi_4+\frac{12}{13}\phi_5 = \frac{84}{13}+R_5; \quad \phi_5=0. \tag{3.19e}$$

We now have a set of equations which can be solved directly if we take them in reverse order. Starting with (3.19e) we have $R_5 = -84/13$, since $\phi_5 = 0$. Knowing ϕ_5 then (3.19d) gives $\phi_4 = 21/13$. Having obtained ϕ_4 and ϕ_5 equation (3.19c) gives $\phi_3 = 49/13$. Then knowing ϕ_3, ϕ_4, ϕ_5 and with ϕ_2 prescribed, (3.19b) gives $R_2 = -46/13$ immediately. Finally we complete the *back substitution* process by determining ϕ_1 from (3.19a) since ϕ_2, ϕ_3, ϕ_4 are known at this stage. This gives $\phi_1 = 179/13$. Since the above procedure is quite systematic it can be readily programmed.

The global stiffness matrix must be assembled and the stiffness equations reduced only if the element stiffnesses have been changed for the current iteration. The full assembly and reduction process must be followed if

KRESL = 1, but only the global load vector need be formed and reduced if KRESL = 2. In this way a considerable number of arithmetic operations are avoided if only equation resolution is to be undertaken. This facility is incorporated in the equation solution subroutines presented in the following sections.

The principles discussed in this section can now be repeated as a FORTRAN operation. Four subroutines are presented which undertake the respective tasks of equation assembly, equation reduction by Gaussian direct elimination, the back substitution process and reduction of subsequent load vectors for equation resolution.

3.4.2 Subroutine ASSEMB

This subroutine assembles the element nodal loads to form the global load vector. Also, the contributions of individual elements are assembled to form the global stiffness matrix. The variables employed in the subroutine are listed below and descriptive notes are again provided immediately after the FORTRAN listing.

Dictionary of variable names (with dimensions)

ASLOD (MSVAB)	ASsembled LOaD vector
ASTIF (MSVAB, MSVAB)	Assembled global STIFfness matrix
RLOAD (MEVAB)	Element load vector
ESTIF (MEVAB, MEVAB)	Element STIFfness matrix
IELEM, NELEM, MELEM	Index, Number, Maximum of ELEMents
IFILE	Input FILE
IDOFN, JDOFN, NODFN	Index, Index, Number of Degrees Of Freedom per Node
INODE, JNODE, NNODE, MNODE	Index, Index, Number, Maximum of NODes per Element
ISVAB, JSVAB, MSVAB, NSVAB	Index, Index, Maximum, Number of global Structural VAriaBles
JFILE	Output file
KRESL	Equation resolution index
LNODS (MELEM, MNODE)	ELement NODe numberS listed for each element
NODEI	NODE I
NODEJ	NODE J
NCOLS	Number of the COLumn in the global Structural stiffness matrix
NROWS	Number of the ROW in the global Structural stiffness matrix and load vector

NCOLE	Number of the COLumn in the Element stiffness matrix
NROWE	Number of the ROW in the Element stiffness matrix and load vector
MEVAB	Maximum of Element VAriaBles

```
      SUBROUTINE ASSEMB                                                 ASEM   1
C***********************************************************************ASEM   2
C                                                                       ASEM   3
C *** ELEMENT ASSEMBLY ROUTINE                                          ASEM   4
C                                                                       ASEM   5
C***********************************************************************ASEM   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     ASEM   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     ASEM   8
     .             NITER,NOUTP,FACTO,PVALU                              ASEM   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          ASEM  10
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       ASEM  11
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           ASEM  12
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      ASEM  13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           ASEM  14
C                                                                       ASEM  15
C     ELEMENT ASSEMBLY ROUTINE                                          ASEM  16
C                                                                       ASEM  17
      REWIND 1                                                          ASEM  18
      DO 10 ISVAB=1,NSVAB                                               ASEM  19
   10 ASLOD(ISVAB)=0.0                                                  ASEM  20
      IF(KRESL.EQ.2) GO TO 30                                           ASEM  21
      DO 20 ISVAB=1,NSVAB                                               ASEM  22
      DO 20 JSVAB=1,NSVAB                                               ASEM  23
   20 ASTIF(ISVAB,JSVAB)=0.0                                            ASEM  24
   30 CONTINUE                                                          ASEM  25
C                                                                       ASEM  26
C     ASSEMBLE THE ELEMENT LOADS                                        ASEM  27
C                                                                       ASEM  28
      DO 50 IELEM=1,NELEM                                               ASEM  29
      READ(1) ESTIF                                                     ASEM  30
      DO 40 INODE=1,NNODE                                               ASEM  31
      NODEI=LNODS(IELEM,INODE)                                          ASEM  32
      DO 40 IDOFN=1,NDOFN                                               ASEM  33
      NROWS=(NODEI-1)*NDOFN + IDOFN                                     ASEM  34
      NROWE=(INODE-1)*NDOFN + IDOFN                                     ASEM  35
      ASLOD(NROWS)=ASLOD(NROWS) + ELOAD(IELEM,NROWE)                    ASEM  36
C                                                                       ASEM  37
C     ASSEMBLE THE ELEMENT STIFFNESS MATRICES                           ASEM  38
C                                                                       ASEM  39
      IF(KRESL.EQ.2) GO TO 40                                           ASEM  40
      DO 40 JNODE = 1,NNODE                                             ASEM  41
      NODEJ=LNODS(IELEM,JNODE)                                          ASEM  42
      DO 40 JDOFN =1,NDOFN                                              ASEM  43
      NCOLS=(NODEJ-1)*NDOFN + JDOFN                                     ASEM  44
      NCOLE=(JNODE-1)*NDOFN + JDOFN                                     ASEM  45
      ASTIF(NROWS,NCOLS)=ASTIF(NROWS,NCOLS) + ESTIF(NROWE,NCOLE)        ASEM  46
   40 CONTINUE                                                          ASEM  47
   50 CONTINUE                                                          ASEM  48
      RETURN                                                            ASEM  49
      END                                                               ASEM  50
```

ASEM 18 Rewind file ready for reading the individual element stiffness matrices.

ASEM 19–20 Set the global load vector, ASLOD, to zero.

ASEM 21–25	If only equation resolution is to be performed during this iteration, do not set the global stiffness coefficients to zero.
ASEM 29	Loop for each element.
ASEM 30	Read ESTIF for the current element.
ASEM 31	Loop for each node 'INODE' of current element.
ASEM 32	From LNODS array identify node number of current node 'INODE'.
ASEM 33	Loop for each degree of freedom of the current node 'INODE'.
ASEM 34	Establish the row position in the global stiffness matrix and load vector.
ASEM 35	Establish the row position in the element stiffness matrix and load vector.
ASEM 36	Add the contribution to the global load vector from the element load vector.
ASEM 40	If equation resolution is to be performed, avoid assembling the global stiffness matrix.
ASEM 41	Loop for each node 'JNODE' of the current element.
ASEM 42	From LNODS array identify node number of current node 'JNODE'.
ASEM 43	Loop for each degree of freedom of the current node 'JNODE'.
ASEM 44	Establish the column position in the global stiffness matrix.
ASEM 45	Establish the column position in the element stiffness matrix.
ASEM 46	Add the contribution to the global stiffness matrix from the element stiffness matrix.
ASEM 48	End element loop.

For the problem described in Section 3.4.1, the main variables have the following values

$$\text{NNODE} = 2, \ \text{NELEM} = 4, \ \text{NDOFN} = 1, \ \text{NSVAB} = 5,$$

$$\text{LNODS} = \begin{bmatrix} 1 & 3 \\ 2 & 3 \\ 3 & 4 \\ 4 & 5 \end{bmatrix} \begin{array}{l} \text{— Element I} \\ \text{— Element II} \\ \text{— Element III} \\ \text{— Element IV.} \end{array}$$

3.4.3 Subroutine GREDUC

This subroutine undertakes the equation elimination process for equation solution by Gaussian reduction as outlined in Section 3.4.1. The additional variable names employed are defined below.

Dictionary of variable names

ASLOD (MEQNS)	ASembled LOaD vector.
ASTIF (MEQNS, MEQNS)	Assembled global STIFfness matrix.

IEQNS, NEQNS, MEQNS	Index, Number, Maximum of EQuatioNS.
IFPRE (MEQNS)	Vector of parameters defining the fixity of a node. 0 – free; 1 – fixed.
FIXED (MEQNS)	Vector of prescribed displacements (zero if not prescribed).
ICOLS	Index COLumn of Structural stiffness matrix.
IROWS	Index ROW of Structural stiffness matrix.
FACTR	Gaussian reduction FACToR.
FRESV ( )	Stored Gaussian reduction factors.
PIVOT	Diagonal term of variable which is currently being eliminated.

```
      SUBROUTINE GREDUC
C***********************************************************************
C
C *** GAUSSIAN REDUCTION ROUTINE
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
C
C     GAUSSIAN REDUCTION ROUTINE
C
      KOUNT=0
      NEQNS=NSVAB
      DO 70 IEQNS=1,NEQNS
      IF(IFPRE(IEQNS).EQ.1) GO TO 40
C
C     REDUCE EQUATIONS
C
      PIVOT=ASTIF(IEQNS,IEQNS)
      IF(ABS(PIVOT).LT.1.0E-10) GO TO 60
      IF(IEQNS.EQ.NEQNS) GO TO 70
      IEQN1=IEQNS+1
      DO 30 IROWS=IEQN1,NEQNS
      KOUNT=KOUNT+1
      FACTR=ASTIF(IROWS,IEQNS)/PIVOT
      FRESV(KOUNT)=FACTR
      IF(FACTR.EQ.0.0) GO TO 30
      DO 10 ICOLS=IEQNS,NEQNS
      ASTIF(IROWS,ICOLS)=ASTIF(IROWS,ICOLS)-FACTR*ASTIF(IEQNS,ICOLS)
   10 CONTINUE
      ASLOD(IROWS)=ASLOD(IROWS)-FACTR*ASLOD(IEQNS)
   30 CONTINUE
      GO TO 70
C
C     ADJUST RHS(LOADS) FOR PRESCRIBED DISPLACEMENTS
```

```
C                                                              GRED  42
   40 DO 50 IROWS=IEQNS,NEQNS                                  GRED  43
      ASLOD(IROWS)=ASLOD(IROWS)-ASTIF(IROWS,IEQNS)*FIXED(IEQNS) GRED  44
   50 CONTINUE                                                 GRED  45
      GO TO 70                                                 GRED  46
   60 WRITE(6,900)                                             GRED  47
  900 FORMAT(5X,15HINCORRECT PIVOT)                            GRED  48
      STOP                                                     GRED  49
   70 CONTINUE                                                 GRED  50
      RETURN                                                   GRED  51
      END                                                      GRED  52
```

GRED 18 Set the counter over the Gaussian reduction factorisation terms to zero.

GRED 19 Set the number of equations to be solved equal to the total number of variables in the structure, NSVAB.

GRED 20 Loop for each equation—this equation is associated with the variable about to be eliminated.

GRED 21 If this variable is fixed, skip to 40.

GRED 25 Extract PIVOT—the leading diagonal term.

GRED 26 Check for zero PIVOT in which case write a message and stop the program.

GRED 27–38 Alter equations below equation 'IEQNS', not those above, according to (3.12). Note that the Gaussian factorisation terms are stored for use during equation resolution.

GRED 43–45 For prescribed variables adjust the R.H.S. (or load) terms according to (3.13).

GRED 47–49 For an invalid pivot value, write a message and terminate execution of the program.

For the problem considered in Section 3.4.1 the main variables have the following values:

$$\text{NEQNS} = 5, \quad \text{ASLOD} = \begin{bmatrix} 10 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \text{modified ASLOD} = \begin{bmatrix} 10 \\ -4 \\ 14 \\ 42/5 \\ 84/13 \end{bmatrix}$$

$$\text{ASTIF} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 \\ -1 & -2 & 6 & -3 & 0 \\ 0 & 0 & -3 & 7 & -4 \\ 0 & 0 & 0 & -4 & 4 \end{bmatrix}, \quad \begin{matrix}\text{modified}\\ \text{ASTIF}\end{matrix} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 5 & -3 & 1 \\ 0 & 0 & 0 & 26/5 & -4 \\ 0 & 0 & 0 & 0 & 12/13 \end{bmatrix}$$

$$\text{IFPRE} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \quad \text{FIXED} = \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The computational effort in this reduction process is proportional to n^3. This can be approximately halved if we take advantage of the symmetry of the stiffness matrices.

3.4.4 Subroutine BAKSUB

The object of this subroutine is to perform the back substitution process required after equation elimination by Gaussian reduction. This results in sequential solution for all the unknowns and reactions at nodal points at which values of the unknown have been prescribed. In the nonlinear solution processes described in Chapter 2, the values of the unknown determined during any iteration may or may not be the total values depending on the solution algorithm being employed. If the method of direct iteration is being used, then, according to equation (2.3), the value of φ determined during any iteration is the total value. For all other solution techniques considered the total values of the unknown are accumulated according to the corrections determined during each iteration, as indicated for example by (2.12).

Therefore, for the direct iteration process, it is simply necessary to transfer the calculated values of the unknowns and the reactions to the arrays TDISP (ISVAB, IDOFN) and TREAC (ISVAB, IDOFN) for output later. This transfer is only necessary to allow the same subroutine to be employed for output of results for all four programs.

Subroutine BAKSUB will now be presented in a form suitable for nonlinear solution dy direct iteration.

Dictionary of variable names

ASLOD (MEQNS)	Reduced load vector.
ASTIF (MEQNS, MEQNS)	Reduced global stiffness matrix.
IEQNS, NEQNS, MEQNS	Index, Number, Maximum of EQatioNS.
IFPRE (MEQNS)	Vector of parameters defining the fixing of a node. 0 – free; 1 – fixed.
FIXED (MEQNS)	Vector of prescribed displacements (zero if not prescribed).
PIVOT	Diagonal term of variable currently being evaluated.
REACT (MEQNS)	REACTions at nodes with prescribed displacements.
XDISP (MEQNS)	Displacement at nodes.

```
      SUBROUTINE BAKSUB                                                 BAKS   1
C***********************************************************************BAKS   2
C                                                                       BAKS   3
C *** BACK-SUBSTITUTION ROUTINE                                         BAKS   4
C                                                                       BAKS   5
C***********************************************************************BAKS   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     BAKS   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     BAKS   8
     .             NITER,NOUTP,FACTO,PVALU                              BAKS   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          BAKS  10
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       BAKS  11
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           BAKS  12
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      BAKS  13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           BAKS  14
C                                                                       BAKS  15
C     BACK-SUBSTITUTION ROUTINE                                         BAKS  16
C                                                                       BAKS  17
      NEQNS=NSVAB                                                       BAKS  18
      DO 10 IEQNS=1,NEQNS                                               BAKS  19
      REACT(IEQNS)=0.0                                                  BAKS  20
   10 CONTINUE                                                          BAKS  21
      NEQN1=NEQNS+1                                                     BAKS  22
      DO 40 IEQNS=1,NEQNS                                               BAKS  23
      NBACK=NEQN1-IEQNS                                                 BAKS  24
      PIVOT=ASTIF(NBACK,NBACK)                                          BAKS  25
      RESID=ASLOD(NBACK)                                                BAKS  26
      IF(NBACK.EQ.NEQNS) GO TO 30                                       BAKS  27
      NBAC1=NBACK+1                                                     BAKS  28
      DO 20 ICOLS=NBAC1,NEQNS                                           BAKS  29
      RESID=RESID-ASTIF(NBACK,ICOLS)*XDISP(ICOLS)                       BAKS  30
   20 CONTINUE                                                          BAKS  31
   30 IF(IFPRE(NBACK).EQ.0) XDISP(NBACK)=RESID/PIVOT                    BAKS  32
      IF(IFPRE(NBACK).EQ.1) XDISP(NBACK)=FIXED(NBACK)                   BAKS  33
      IF(IFPRE(NBACK).EQ.1) REACT(NBACK)=-RESID                         BAKS  34
   40 CONTINUE                                                          BAKS  35
      KOUNT=0                                                           BAKS  36
      DO 50 IPOIN=1,NPOIN                                               BAKS  37
      DO 50 IDOFN=1,NDOFN                                               BAKS  38
      KOUNT=KOUNT+1                                                     BAKS  39
      TDISP(IPOIN,IDOFN)= XDISP(KOUNT)                                  BAKS  40
   50 TREAC(IPOIN,IDOFN)= REACT(KOUNT)                                  BAKS  41
      RETURN                                                            BAKS  42
      END                                                               BAKS  43
```

BAKS 19–21 Zero space for reactions.
BAKS 22–24 Loop backwards over each equation.
BAKS 25 Use the same PIVOT as in subroutine GREDUC.
BAKS 27 For the last equation (the first to be solved) we do not have any other variables to substitute (i.e. bypass the loop).
BAKS 28–31 Evaluate RESID from previously calculated variables.
BAKS 32 If the variable is not prescribed evaluate the variable.
BAKS 34 If the variable is prescribed evaluate the R.H.S. reaction.
BAKS 36–41 Store the solved variables and reactions in new arrays for output.

For the problem described in Section 3.4.1, the arrays employed in addition to those utilised in Subroutine GREDUC have the following values:

$$\text{TDISP} = \text{XDISP} = \begin{bmatrix} 179/13 \\ 2 \\ 49/13 \\ 21/13 \\ 0 \end{bmatrix}, \quad \text{TREAC} = \text{REACT} = \begin{bmatrix} 0 \\ -46/13 \\ 0 \\ 0 \\ -84/13 \end{bmatrix}.$$

It should be noted that nonzero reactions are obtained only for nodal positions at which the value of the unknown has been prescribed. For the Newton–Raphson, Tangential Stiffness and Initial Stiffness methods, the calculated unknowns and reactions must be accumulated from the values obtained during each iteration. Therefore, for these applications, statements BAKS 36–41 in the above listing must be replaced by

```
      KOUNT=0                                                 BAKS  36
      DO 50 IPOIN=1,NPOIN                                     BAKS  37
      DO 50 IDOFN=1,NDOFN                                     BAKS  38
      KOUNT=KOUNT+1                                           BAKS  39
      TDISP(IPOIN,IDOFN)= TDISP(IPOIN,IDOFN)+XDISP(KOUNT)     BAKS  40
   50 TREAC(IPOIN,IDOFN)= TREAC(IPOIN,IDOFN)+REACT(KOUNT)     BAKS  41
```

with the arrays TDISP and TREAC being initially set to zero at the beginning of the program.

For these three solution algorithms a final further programming addition must be made. When determining the residual forces according to (2.4), the contribution to f of the reactions at nodal points at which the value of the unknown is prescribed must be accounted for, since any reactions can be interpreted as additional applied loads necessary to maintain the prescribed value of the unknown. Therefore, the evaluated reactions must be added into the vector of applied nodal loads at every iteration. This task can be accomplished by the following coding inserted immediately before the RETURN statement:

```
      DO 90 IPOIN=1,NPOIN                                     BAKS  42
      DO 60 IELEM=1,NELEM                                     BAKS  43
      DO 60 INODE=1,NNODE                                     BAKS  44
      NLOCA=LNODS(IELEM,INODE)                                BAKS  45
   60 IF(IPOIN.EQ.NLOCA) GO TO 70                             BAKS  46
   70 DO 80 IDOFN=1,NDOFN                                     BAKS  47
      NPOSN=(IPOIN-1)*NDOFN+IDOFN                             BAKS  48
      IEVAB=(INODE-1)*NDOFN+IDOFN                             BAKS  49
   80 TLOAD(IELEM,IEVAB)=TLOAD(IELEM,IEVAB)+REACT(NPOSN)      BAKS  50
   90 CONTINUE                                                BAKS  51
```

BAKS 42 Loop over each nodal point.

BAKS 43–46 Search through the element nodal connections until one is found corresponding to the nodal point currently under consideration. As soon as one is found, abandon the search. Note that it is immaterial in which element the node is found since all element contributions will be finally assembled.

BAKS 47–50 Add the nodal reaction into the appropriate position in the array of applied element loads.

3.4.5 Subroutine RESOLV

As stated in Section 3.4.1, for equation resolution (indicated by KRESL = 2) only the global load vector need be formed and reduced. Subroutine RESOLV merely reduces the R.H.S. (or load) terms by standard Gaussian elimination using the same operations as employed in Subroutine GREDUC, Section 3.4.3. The Gaussian factorisation terms were evaluated and stored in GREDUC and are now utilised in this subroutine. The programming logic follows that of Subroutine GREDUC and can be readily understood by reference to Section 3.4.3.

```
      SUBROUTINE RESOLV
C***********************************************************************
C
C *** RESOLVING GAUSSIAN REDUCTION ROUTINE
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      KOUNT=0
      NEQNS=NSVAB
      DO 40 IEQNS=1,NEQNS
      IF(IFPRE(IEQNS).EQ.1) GO TO 20
C
C     REDUCE RHS
C
      IF(IEQNS.EQ.NEQNS) GO TO 40
      IEQN1=IEQNS+1
      DO 10 IROWS=IEQN1,NEQNS
      KOUNT=KOUNT+1
      FACTR=FRESV(KOUNT)
      IF(FACTR.EQ.0) GO TO 10
      ASLOD(IROWS)=ASLOD(IROWS)-FACTR*ASLOD(IEQNS)
   10 CONTINUE
      GO TO 40
C
C     ADJUST RHS TO PRESCRIBED DISPLACEMENTS
C
   20 DO 30 IROWS=IEQNS,NEQNS
      ASLOD(IROWS)=ASLOD(IROWS)-ASTIF(IROWS,IEQNS)*FIXED(IEQNS)
   30 CONTINUE
   40 CONTINUE
      RETURN
      END
```

3.4.6 Improved numerical algorithm for equation solution

Substantial economies can be achieved in both core storage requirements and execution times if advantage is taken of the banded symmetric form of the global stiffness matrix. Since:

- By recognising that the global stiffness matrix is symmetric, it is necessary only to store the upper (or lower) triangular part of the stiffness matrix.
- By noting that all the non-zero coefficients in the global stiffness matrix occur in a band adjacent to the leading diagonal, further reductions in the core storage requirements can be made, as well as a significant reduction in the number of arithmetic operations undertaken in the equation reduction and backsubstitution phases.

In order to introduce these enhancements it is convenient to store the global stiffness matrix as a one-dimensional array. The necessary programming changes required to the subroutines presented in Sections 3.4.2–3.4.5 are fully documented in Ref. 5.

3.5 Output of results

The next subroutine common to all four programs presented is subroutine RESULT whose function is to output the results at a frequency governed by a parameter input in Subroutine INCLOD described in Section 3.7. In order to make the subroutine applicable to all four cases, quantities will be output for some situations which are physically meaningless. In particular for quasi-harmonic problems, output items termed *stress* and *plastic or nonlinear strain* are output as zero values for this reason. For nonlinear elastic problems the latter term is the total strain, ϵ, defined in Section 2.4 and for elasto-plastic situations it is the plastic strain component, ϵ_p, defined in Section 2.5. For both cases the stress quantity output is the axial stress existing in each constant stress element employed.

Subroutine RESULT will now be listed.

```
      SUBROUTINE  RESULT                                                RSLT
C***********************************************************************RSLT
C                                                                       RSLT
C *** OUTPUTS DISPLACEMENT , REACTIONS AND STRESSES                     RSLT
C                                                                       RSLT
C***********************************************************************RSLT
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     RSLT
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     RSLT
     .             NITER,NOUTP,FACTO,PVALU                              RSLT
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          RSLT
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       RSLT
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           RSLT
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      RSLT
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           RSLT
      IF(NDOFN.EQ.1) WRITE(6,900)                                       RSLT
```

```
900 FORMAT(1H0,5X,'NODE',4X,'DISPL.',12X,'REACTIONS')                  RSLT 16
    IF(NDOFN.EQ.2) WRITE(6,910)                                        RSLT 17
910 FORMAT(1H0,5X,'NODE',4X,'DISPL.',12X,'REACTION',                   RSLT 18
   .         7X,'DISPL.',12X,'REACTION')                               RSLT 19
    DO 10 IPOIN=1,NPOIN                                                RSLT 20
 10 WRITE(6,920) IPOIN,(TDISP(IPOIN,IDOFN),TREAC(IPOIN,IDOFN),         RSLT 21
   .IDOFN=1,NDOFN)                                                     RSLT 22
920 FORMAT(I10,2(E14.6,5X,E14.6))                                      RSLT 23
    IF(NDOFN.EQ.2) WRITE(6.930)                                        RSLT 24
930 FORMAT(1H0,2X,'ELEMENT',12X,'STRESSES',12X,'PL.STRAIN')            RSLT 25
    IF(NDOFN.EQ.1) WRITE(6.940)                                        RSLT 26
940 FORMAT(1H0,2X,'ELEMENT',5X,'STRESSES',5X,'PL.STRAIN')              RSLT 27
    DO 20 IELEM=1,NELEM                                                RSLT 28
 20 WRITE(6,950) IELEM,(STRES(IELEM,IDOFN),IDOFN=1,NDOFN),             RSLT 29
   .             PLAST(IELEM)                                          RSLT 30
950 FORMAT(I10,3E14.6)                                                 RSLT 31
    RETURN                                                             RSLT 32
    END                                                                RSLT 33
```

RSLT 15–23 Write titles and output the calculated unknown and reaction at each nodal point. Non-zero reactions are only obtained for nodal points at which the value of the unknown is prescribed.

RSLT 24–31 Write titles and output the stress and plastic or nonlinear elastic strain for each element.

Note that provision is made for output of results for the beam bending application of Chapter 5.

3.6 Subroutine INITAL

The function of this subroutine is to initialise to zero some arrays used by other subroutines.

```
      SUBROUTINE INITAL
C***********************************************************************
C
C *** INITIALIZES TO ZERO ALL ACCUMULATIVE ARRAYS
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      DO 20 IELEM=1,NELEM
      PLAST(IELEM)=0.0
      DO 10 IDOFN=1,NDOFN
```

```
   10 STRES(IELEM,IDOFN)=0.0          INTL 18
      DO 20 IEVAB=1,NEVAB             INTL 19
      ELOAD(IELEM,IEVAB)=0.0          INTL 20
   20 TLOAD(IELEM,IEVAB)=0.0          INTL 21
      DO 30 IPOIN=1,NPOIN             INTL 22
      DO 30 IDOFN=1,NDOFN             INTL 23
      TDISP(IPOIN,IDOFN)=0.0          INTL 24
   30 TREAC(IPOIN,IDOFN)=0.0          INTL 25
      RETURN                          INTL 26
      END                             INTL 27
```

INTL 15–18 Initialise to zero the plastic or nonlinear strain vector and the stress vector.

INTL 20 Initialise the array, ELOAD, which will contain the out of balance loading to be applied in solution for any iteration. For techniques other than the direct iteration method, this vector will contain the residual nodal forces and thus differs from the vector of applied loads.

INTL 21 Initialise the vector of applied loads.

INTL 22–25 Initialise the vector of total unknowns and total reactions to zero.

3.7 Load increment subroutine, INCLOD

This subroutine controls the incrementing of the applied loads. For each increment of load, data is input to this segment to control the upper limit to the number of iterations, the output frequency, the size of load increment and the convergence tolerance limit. These quantities are specifically input as:

NITER Maximum permissible number of iterations. This is a safety measure to cover situations where the solution process does not converge. After performing NITER iteration cycles the program will then stop.

NOUTP This parameter controls the frequency of output of results. In order to examine the iterative procedure the user may wish to obtain results at stages other than the converged solution.

- 0 – Print the results on convergence to the nonlinear solution only, for each load increment.
- 1 – Print the results after the first iteration *and* after convergence for each load increment.
- 2 – Print the results after every iteration for each load increment.

FACTO This quantity controls the magnitude of any load increment. The applied loading is input in subroutine DATA into the array RLOAD as described in Section 3.2. The size of any load increment is then defined to be FACTO*RLOAD

(IELEM, INODE) with the increment size factor, FACTO, being input for each increment. This permits unequal load increments to be taken. It should be noted that the applied loading at any instant is accumulative. Therefore, if FACTO is input for the first three increments as respectively 0·5, 0·3 and 0·1, the total loading applied to the structure during the third increment is 0·9 times the loading input in subroutine DATA. The above also holds for loading by incremental prescribed displacements.

TOLER This item of data controls the tolerance permitted on the convergence process. Its use will be described in detail in Sections 3.9.2 and 3.9.3.

Subroutine INCLOD is now presented and described:

```
      SUBROUTINE  INCLOD                                                INCL   1
C***********************************************************************INCL   2
C                                                                       INCL   3
C *** INPUTS DATA FOR CURRENT INCREMENT AND UPDATES LOAD VECTOR         INCL   4
C                                                                       INCL   5
C***********************************************************************INCL   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     INCL   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     INCL   8
     .             NITER,NOUTP,FACTO,PVALU                              INCL   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          INCL  10
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       INCL  11
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           INCL  12
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      INCL  13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           INCL  14
      READ (5,900) NITER,NOUTP,FACTO,TOLER                              INCL  15
  900 FORMAT(2I5,2F15.5)                                                INCL  16
      WRITE(6,905) IINCS,NITER,NOUTP,FACTO,TOLER                        INCL  17
  905 FORMAT(1H0,5X,'IINCS =',I5,3X,'NITER =',I5,3X,'NOUTP =',I5,       INCL  18
     .       3X,'FACTO =',E14.6,3X,'TOLER =',E14.6)                     INCL  19
      DO 10 IELEM=1,NELEM                                               INCL  20
      DO 10 IEVAB=1,NEVAB                                               INCL  21
      ELOAD(IELEM,IEVAB)=ELOAD(IELEM,IEVAB)+RLOAD(IELEM,IEVAB)*FACTO    INCL  22
      TLOAD(IELEM,IEVAB)=TLOAD(IELEM,IEVAB)+RLOAD(IELEM,IEVAB)*FACTO    INCL  23
   10 CONTINUE                                                          INCL  24
      RETURN                                                            INCL  25
      END                                                               INCL  26
```

INCL 15–19 Read and write the input data required for each load increment as described previously in this section.

INCL 20–24 Add the current increment of load into the out of balance load array ELOAD and the total applied load vector TLOAD.

3.8 The master or controlling segment

The final portion of the program which will be common to all four programs (subject to the minor differences indicated in Fig. 3.1) is the master segment which controls the calling, in order, of the other subroutines. This program segment also controls the iterative process and also the incrementing of the applied loads, where appropriate.

The following channel numbers are employed by the programs: 5 (card reader), 6 (line printer), 1 (scratch file).

The MASTER segment will now be presented in the form required in the next section for the solution of one-dimensional quasi-harmonic problems by direct iteration. For other applications it is only necessary to arrange for the calling of appropriate subroutines as indicated in Fig. 3.1.

```
      MASTER UNIDIM                                                     QUIT   1
C***********************************************************************QUIT   2
C                                                                       QUIT   3
C *** PROGRAM FOR THE 1-D SOLUTION OF NONLINEAR PROBLEMS                QUIT   4
C                                                                       QUIT   5
C***********************************************************************QUIT   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     QUIT   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     QUIT   8
     .             NITER,NOUTP,FACTO,PVALU                              QUIT   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          QUIT  10
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       QUIT  11
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           QUIT  12
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      QUIT  13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           QUIT  14
      CALL  DATA                                                        QUIT  15
      CALL  INITAL                                                      QUIT  16
      DO 30 IINCS=1,NINCS                                               QUIT  17
      CALL  INCLOD                                                      QUIT  18
      DO 10 IITER=1,NITER                                               QUIT  19
      CALL  NONAL                                                       QUIT  20
      IF(KRESL.EQ.1) CALL  STIFF1                                       QUIT  21
      CALL  ASSEMB                                                      QUIT  22
      IF(KRESL.EQ.1) CALL GREDUC                                        QUIT  23
      IF(KRESL.EQ.2) CALL RESOLV                                        QUIT  24
      CALL  BAKSUB                                                      QUIT  25
      CALL  MONITR (RINTL)                                              QUIT  26
      IF(NCHEK.EQ.0) GO TO 20                                           QUIT  27
      IF(IITER.EQ.1.AND.NOUTP.EQ.1) CALL  RESULT                        QUIT  28
      IF(NOUTP.EQ.2) CALL  RESULT                                       QUIT  29
   10 CONTINUE                                                          QUIT  30
      WRITE(6,900)                                                      QUIT  31
  900 FORMAT(1H0,5X,'SOLUTION NOT CONVERGED')                           QUIT  32
      STOP                                                              QUIT  33
   20 CALL  RESULT                                                      QUIT  34
   30 CONTINUE                                                          QUIT  35
      STOP                                                              QUIT  36
      END                                                               QUIT  37
```

QUIT 15	Call the subroutine which reads the input data as described in Section 3.2.
QUIT 16	Call the subroutine which initialises various arrays to zero.
QUIT 17	Enter the DO LOOP over the number of load increments.
QUIT 18	Call the subroutine which increments the applied loads.
QUIT 19	Enter the DO LOOP over the maximum permissible number of iterations.
QUIT 20	Call the subroutine which controls the solution process as described in Section 3.3.
QUIT 21	If the element stiffnesses are to be reformulated, call the appropriate subroutine.

QUIT 22–25	Call the subroutines which assemble the element stiffnesses and solve for the unknowns and reactions.
QUIT 26	Call the subroutine which monitors the convergence process. This subroutine differs for the direct iteration method from that for the three other cases.
QUIT 27	If the solution has converged, abandon the iterative process.
QUIT 28–29	Output the results according to the display code, NOUTP, supplied as input for this particular load increment.
QUIT 31–33	If the solution procedure reaches the maximum number of iterations permitted without convergence occurring, write a message and stop the program.
QUIT 34	Otherwise output the converged results.
QUIT 35	Return to process the next increment of load.

3.9 Program for the solution of one-dimensional quasi-harmonic problems by direct iteration

We now assemble a computer program which permits the solution of one-dimensional problems governed by a nonlinear quasi-harmonic equation. The behaviour of several physical situations can be described by such a model and some numerical examples will be provided at the end of this section.

Most of the subroutines required for this program have been already described in the preceding sections of this chapter and, in particular, the master segment which controls the entire numerical process was described in Section 3.8. The additional subroutines, pertinent only to this application which must be developed, are the element stiffness generation subroutine, STIFF1, and the solution convergence monitoring subroutine, MONITR. Detailed 'user instructions', listing the required input data, are included in Appendix I.

3.9.1 Element stiffness subroutine, STIFF1

The purpose of this subroutine is to formulate the stiffness matrix for each element in turn and store this data on a disc file. For solution by the method of direct iteration, the stiffness matrix for a one-dimensional element with a linear variation of the unknown is given by equation (2.25). The term K is, however, a specified function of the unknown or its derivatives which must be accounted for when formulating the element stiffnesses for each iteration of the solution sequence. In particular, K is assumed to vary according to

$$K = K_0 f\left(\phi, \ \frac{d\phi}{dx}\right), \tag{3.20}$$

where K_0 is a reference value of K and is specified as material property PROPS (NUMAT, 1) in subroutine DATA. The function $f(\phi, d\phi/dx)$ is defined by means of a FORTRAN FUNCTION statement and must be appropriately specified for each application.

Subroutine STIFFI is now presented and descriptive notes provided.

```
      SUBROUTINE  STIFF1                                                STF1   1
C***********************************************************************STF1   2
C                                                                       STF1   3
C *** CALCULATES ELEMENT STIFFNESS MATRICES                             STF1   4
C                                                                       STF1   5
C***********************************************************************STF1   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE.IINCS,IITER,     STF1   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     STF1   8
     .             NITER,NOUTP,FACTO,PVALU                              STF1   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          STF1  10
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       STF1  11
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           STF1  12
     .             TDISP(26,2),TREAC(26.2),ASTIF(52,52),ASLOD(52),      STF1  13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           STF1  14
      REWIND 1                                                          STF1  15
      DO 10 IELEM=1,NELEM                                               STF1  16
      LPROP=MATNO(IELEM)                                                STF1  17
      STERM=PROPS(LPROP,1)                                              STF1  18
      NODE1=LNODS(IELEM,1)                                              STF1  19
      NODE2=LNODS(IELEM,2)                                              STF1  20
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))                              STF1  21
      AVERG=(TDISP(NODE1,1)+TDISP(NODE2,1))/2.0                         STF1  22
      FMULT=STERM*VARIA(AVERG)/ELENG                                    STF1  23
      ESTIF(1,1)=FMULT                                                  STF1  24
      ESTIF(1,2)=-FMULT                                                 STF1  25
      ESTIF(2,1)=-FMULT                                                 STF1  26
      ESTIF(2,2)=FMULT                                                  STF1  27
      WRITE(1) ESTIF                                                    STF1  28
   10 CONTINUE                                                          STF1  29
      RETURN                                                            STF1  30
      END                                                               STF1  31
```

STF1 15	Rewind the file on which the stiffness matrix for each element will be stored in sequence.
STF1 16	Loop over each element.
STF1 17	Identify the material property of each element.
STF1 18	Set STERM equal to K_0.
STF1 19–20	Identify the node numbers of the element.
STF1 21	Calculate the element length.
STF1 22	Calculate the element temperature as the average of the nodal values.
STF1 23	Calculate the temperature gradient.
STF1 24–27	Compute the components of the element stiffness matrix according to (2.25) with the function $f(\phi, d\phi/dx)$ being VARIA (AVERG).
STF1 28	Write the element stiffness matrix on to disc file.
STF1 29	Termination of DO LOOP over each element.

The function $f(\phi, d\phi/dx)$ must be defined for each application. Below we show, for example, the appropriate function for the variation $K = K_0(1+10\phi)$.

```
      FUNCTION VARIA(AVERG)
C****
C     MULTIPLYING FUNCTION FOR QUASI-HARMONIC STIFFNESS VARIATION
C****
      VARIA=1.0+10.0*AVERG
      RETURN
      END
```

3.9.2 Solution convergence monitoring subroutine, MONITR

Convergence of the numerical process to the nonlinear solution must be monitored by comparing, in some way, the values of the unknowns φ determined during each iteration. One possible method is to compare each individual nodal value with the corresponding value obtained on the previous iteration. Then, provided that this change is negligibly small for all nodal points, convergence can be deemed to have occurred. In this chapter we will employ a *global* convergence check rather than such a *local* one. We will assume that the numerical process has converged if

$$\frac{\left|\sqrt{\left[\sum_{i=1}^{N}(\phi_i{}^r)^2\right]}-\sqrt{\left[\sum_{i=1}^{N}(\phi_i{}^{r-1})^2\right]}\right|}{\sqrt{\left[\sum_{i=1}^{N}(\phi_i{}^1)^2\right]}}\times 100 \leqslant \text{TOLER}, \tag{3.21}$$

where N denotes the total number of nodal points in the problem and $r-1$ and r denote successive iterations. It is assumed that the positive root is always considered and $|\;|$ signifies the absolute value of the numerator. The multiplication factor of 100 on the left-hand side allows the specified tolerance factor TOLER to be considered as a percentage term. Equation (3.21) states that convergence is assumed to have occurred if the difference in the norm of the unknowns between two successive iterations is less than or equal to TOLER times the norm of the unknowns on the first iteration. In practical situations a value of TOLER $= 1{\cdot}0$ (i.e., 1%) is found to be adequate for the majority of applications. Convergence of the solution is indicated by the parameter NCHEK. A value of NCHEK $= 1$ indicates that convergence has not yet occurred, whereas NCHEK $= 0$, denotes a converged solution. Subroutine MONITR is now presented and descriptive notes provided.

```
      SUBROUTINE MONITR(RINTL)
C*********************************************************************
C
C *** CHECKS FOR SOLUTION CONVERGENCE
C
C*********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
```

```
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           MNTR 12
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      MNTR 13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           MNTR 14
      NCHEK=0                                                           MNTR 15
      RCURR=0.0                                                         MNTR 16
      DO 10 IPOIN=1,NPOIN                                               MNTR 17
   10 RCURR=RCURR+TDISP(IPOIN,1)*TDISP(IPOIN,1)                         MNTR 18
      IF(IITER.EQ.1) RINTL=RCURR                                        MNTR 19
      IF(IITER.EQ.1) NCHEK=1                                            MNTR 20
      IF(IITER.EQ.1) GO TO 20                                           MNTR 21
      RATIO=100.0*SQRT(ABS(RCURR-PVALU))/SQRT(RINTL)                    MNTR 22
      IF(RATIO.GT.TOLER) NCHEK=1                                        MNTR 23
   20 PVALU=RCURR                                                       MNTR 24
      WRITE(6,900) NCHEK,RATIO                                          MNTR 25
  900 FORMAT(1H0,5X,18HCONVERGENCE CODE =,I4,3X,28HNORM OF RESIDUAL SUM MNTR 26
     .RATIO =,E14.6)                                                    MNTR 27
      RETURN                                                            MNTR 28
      END                                                               MNTR 29
```

MNTR 15 Set the indicator monitoring convergence to zero. If convergence has not yet occurred this will be set to 1 later in the subroutine.

MNTR 16–18 Compute the norm of the unknowns

$$\sum_{i=1}^{N} \phi_i^2,$$

for the current iteration.

MNTR 19 For the first iteration only compute the denominator of (3.21).

MNTR 20–21 Convergence cannot possibly have occurred on the first iteration, therefore set NCHEK = 1 and skip the remainder of the checking procedure by going to 20.

MNTR 22 Compute the left-hand side of (3.21).

MNTR 23 If (3.21) is not satisfied (i.e., convergence not taken place), set NCHEK = 1.

MNTR 24 Store the current value of the norm of the unknowns for use as

$$\sum_{i=1}^{N} (\phi_i^{r-1})^2$$

during the next iteration.

MNTR 25–27 Output the value of NCHEK and the left-hand side of (3.21).

3.9.3 Numerical examples

The first numerical example considered is illustrated in Fig. 3.3. The situation shown could physically represent the diffusion of a gas through a membrane in which case ϕ is the gas concentration and K is the diffusivity of the membrane. Alternatively, the problem also represents the conduction of heat through a one-dimensional solid in which case ϕ is the temperature and K the thermal conductivity. The boundary conditions assumed are

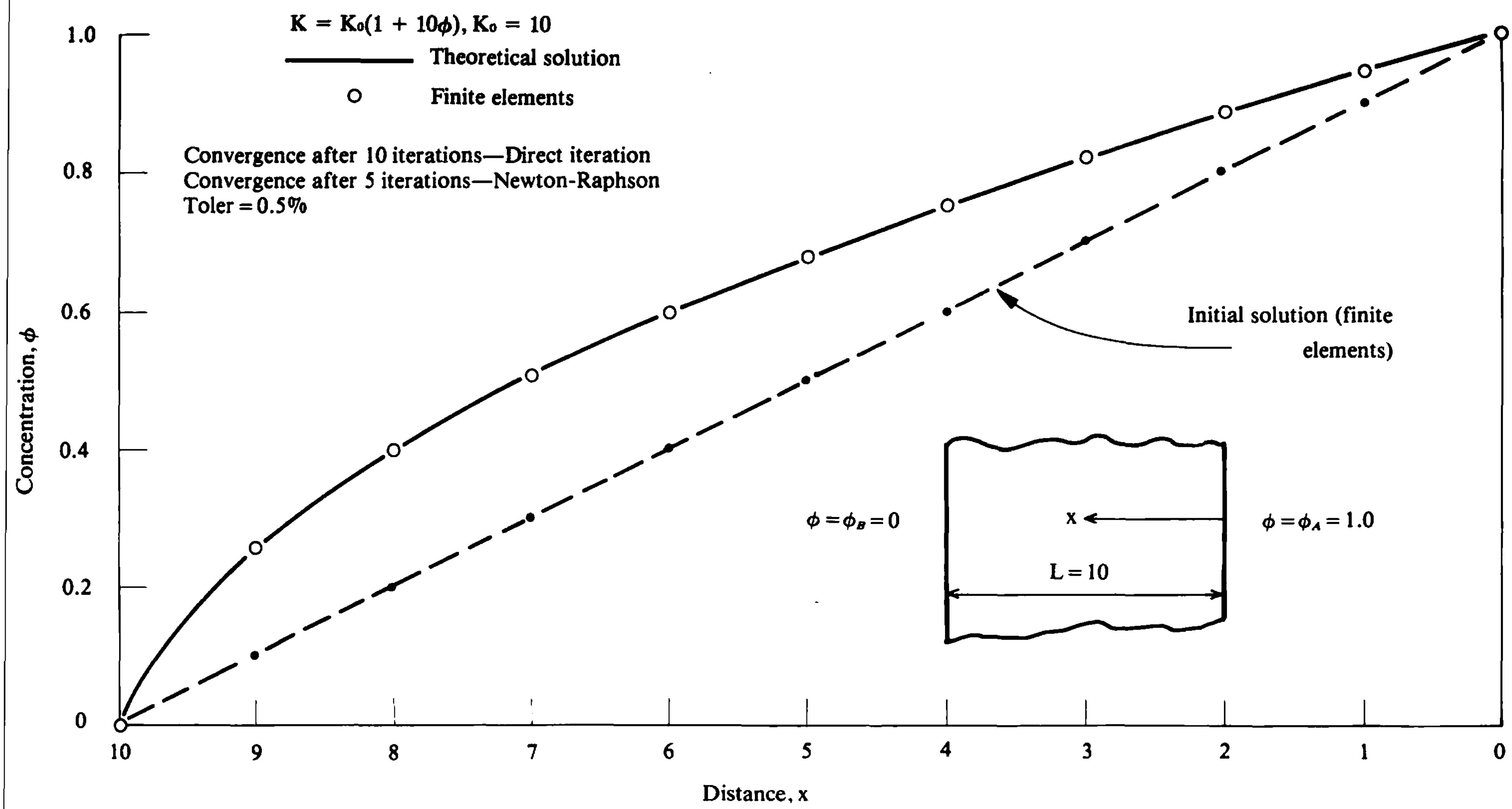


Fig. 3.3 Quasi-harmonic equation example—Problem of gas diffusion through a permeable membrane.

specified values of the unknown at the two boundaries. The term K is assumed to vary with the unknown ϕ according to

$$K = K_0(1+10\phi) = K_0(1+g(\phi)). \tag{3.22}$$

An analytical solution[6] exists for this problem which enables ϕ to be determined from

$$\frac{\phi_A+F(\phi_A)-\phi-F(\phi)}{\phi_A+F(\phi_A)-\phi_B-F(\phi_B)} = \frac{x}{L}, \tag{3.23}$$

where

$$F(\phi) = \int_0^{\phi} g(\phi')\,d\phi'. \tag{3.24}$$

In the present case, $g(\phi) = 10\phi$ which gives on substitution in (3.24) and then in (3.23)

$$\frac{6-\phi-5\phi^2}{6} = \frac{x}{10}, \tag{3.25}$$

which allows ϕ to be determined for any value of x and is shown as the full line in Fig. 3.3. The initial finite element solution (i.e., after the first iteration) is shown in Fig. 3.3 as the broken line and, as expected, is linear. The results upon convergence, after 10 iterations, of the process are then included as circles and it is seen that the numerical solution coincides with the theoretical values. For example, for $x = 6$, the theoretical solution is $\phi = 0{\cdot}6$, whilst the finite element analysis yieds $\phi = 0{\cdot}599999$ (see Appendix IV).

The second example considered includes the effect of the term Q in (2.15). For thermal problems this can be physically interpreted as a heat generation/unit length and must be specified as a loading, according to (2.26), in subroutine DATA. Figure 3.4 shows the problem to be considered. A bar with its surface insulated generates heat internally and the temperature at its ends is maintained at zero value. Due to symmetry only one half of the problem is analysed with the symmetry condition $d\phi/dx = 0$ at the centreline being invoked. The initial solution corresponding to $K = K_0$ is shown and is practically identical to the theoretical value. The process converged to the nonlinear solution after 12 iterations with the temperature being markedly reduced. The reduction is greater in regions of higher initial temperature due to the comparatively greater increase in material 'stiffness' in these areas.

3.10 Program for the solution of one-dimensional quasi-harmonic problems by the Newton–Raphson method

As seen in Section 2.3, use of this method results in the assembled stiffness equations being nonsymmetric. The equation assembly and solution routines developed in Section 3.4 made no use of the symmetry properties of the

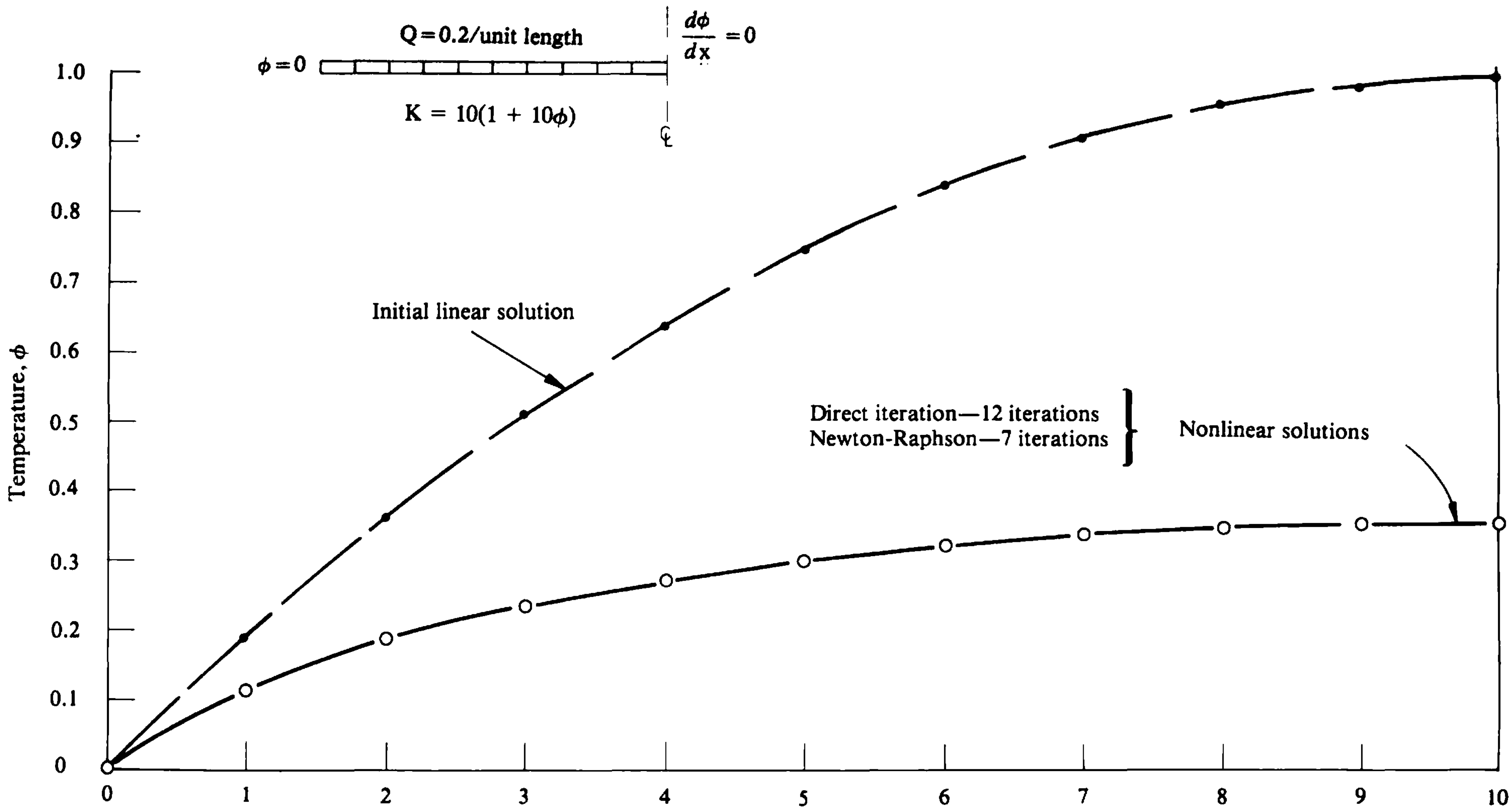


Fig. 3.4 Quasi-harmonic equation example—Heat generation in an axial bar.

stiffness matrices. They are therefore applicable to this method of analysis without modification.

Three additional subroutines need to be developed. These are the element stiffness subroutine ASTIF1 and, since solution convergence is now based on the elimination of the residual forces, subroutine REFOR1 must be formed to calculate these forces and subroutine CONVER to monitor their convergence to zero. The master segment controlling the solution process is again that developed in Section 3.8 and the remaining subroutines accessed by this segment have also been described previously.

3.10.1 Element stiffness formulation subroutine, ASTIF1

For solution by the Newton–Raphson process, the 'stiffness' equations which require solution are summarised in (2.12) where it is seen that the total stiffness is the sum of symmetric, $\boldsymbol{H}$, and nonsymmetric, $\boldsymbol{H}'$, contributions. The symmetric stiffness matrix is given by (2.25) and the nonsymmetric terms depend on the particular form of material nonlinearity. For a material nonlinearity of the form (2.27), the nonsymmetric portion of the stiffness matrix is given by (2.29). The subroutine which evaluates and sums these separate contributions is now presented below.

```
      SUBROUTINE  ASTIF1
C***********************************************************************
C
C *** CALCULATES ELEMENT STIFFNESS MATRICES
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      REWIND 1
      DO 10 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      STERM=PROPS(LPROP,1)
      GRADU=PROPS(LPROP,2)
      NODE1=LNODS(IELEM,1)
      NODE2=LNODS(IELEM,2)
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))
      AVERG=(TDISP(NODE1,1)+TDISP(NODE2,1))/2.0
      FMULT=STERM*VARIA(AVERG)/ELENG
      DIFFR=TDISP(NODE1,1)-TDISP(NODE2,1)
      COEFF=STERM*GRADU*DIFFR/(2.0*ELENG)
      ESTIF(1,1)=FMULT+COEFF
      ESTIF(1,2)=-FMULT+COEFF
      ESTIF(2,1)=-FMULT-COEFF
      ESTIF(2,2)=FMULT-COEFF
      WRITE(1) ESTIF
   10 CONTINUE
      RETURN
      END
```

ASTF 15 Rewind the file on which the stiffness matrix of each element will be stored.
ASTF 16 Loop over each element.
ASTF 17 Identify the material property of each element.
ASTF 18 Set STERM equal to K_0 in (2.27).
ASTF 19 Set GRADU equal to b in (2.27).
ASTF 20–21 Identify the node numbers of the element.
ASTF 22 Calculate the element length.
ASTF 23 Calculate the element temperature as the average of the nodal values.
ASTF 24 Calculate the multiplying term in (2.25) by use of FUNCTION statement VARIA.
ASTF 25–26 Evaluate the multiplying term in (2.29).
ASTF 27–30 Compute the components of the total stiffness matrix.
ASTF 31 Write the element stiffness matrix on to disc file.
ASTF 32 Termination of DO LOOP over each element.

3.10.2 Residual force calculation subroutine REFOR1

The residual forces after any step of the process are obtained from (2.4). The applied nodal forces, f, are known and it only remains to evaluate the 'equivalent nodal forces', $\boldsymbol{H\varphi}$, which are the nodal forces consistent with the unknowns, $\boldsymbol{\varphi}$. It should be noted that $\boldsymbol{H}$ is the linear symmetric matrix defined in (2.25). The equivalent nodal forces at the nodes 1 and 2 of the linear element can be explicitly written, using (2.25), as

$$f_1 = \frac{K}{L}(\phi_1 - \phi_2),$$

$$f_2 = -\frac{K}{L}(\phi_1 - \phi_2). \tag{3.26}$$

The subroutine which evaluates these forces for each element is now presented.

```
      SUBROUTINE REFOR1
C**********************************************************************
C
C *** CALCULATES INTERNAL EQUIVALENT NODAL FORCES
C
C**********************************************************************
      COMMON/UNIM1/NPOIN.NELEM,NBOUN,NLOAD,NPROP,NNODE.IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO.NSVAB,NDOFN,NINCS.NEVAB,
     .             NITER,NOUTP,FACTO.PVALU
      COMMON/UNIM2/PROPS(5.4),COORD(26),LNODS(25,2).IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .      .      MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26.2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4.4)
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
```

```
10 ELOAD(IELEM,IEVAB)=0.0                                           RFR1  17
   DO 20 IELEM=1,NELEM                                              RFR1  18
   LPROP=MATNO(IELEM)                                               RFR1  19
   STERM=PROPS(LPROP,1)                                             RFR1  20
   NODE1=LNODS(IELEM,1)                                             RFR1  21
   NODE2=LNODS(IELEM,2)                                             RFR1  22
   ELENG=ABS(COORD(NODE1)-COORD(NODE2))                             RFR1  23
   AVERG=(TDISP(NODE1,1)+TDISP(NODE2,1))/2.0                        RFR1  24
   STIFF=STERM*VARIA(AVERG)/ELENG                                   RFR1  25
   ELOAD(IELEM,1)= STIFF*(TDISP(NODE1,1)-TDISP(NODE2,1))            RFR1  26
20 ELOAD(IELEM,2)=-STIFF*(TDISP(NODE1,1)-TDISP(NODE2,1))            RFR1  27
   RETURN                                                           RFR1  28
   END                                                              RFR1  29
```

RFR1 15–17 Initialise to zero the array in which the equivalent nodal forces for each element will be stored.

RFR1 18 Loop over each element.

RFR1 19 Identify the material property of each element.

RFR1 20 Set STERM equal to K_0 in (2.27).

RFR1 21–22 Identify the node numbers of the element.

RFR1 23 Calculate the element length.

RFR1 24 Calculate the element temperature as the average of the nodal values.

RFR1 25 Calculate the multiplying term in (2.25).

RFR1 26–27 Compute the equivalent nodal forces according to (3.26).

3.10.3 Solution convergence monitoring subroutine, CONUND

This subroutine must essentially differ from subroutine MONITR described in Section 3.9.2 since convergence is now based on the residual force values rather than values of the unknowns. The convergence criterion employed is similar to that described in (3.21) and is

$$\frac{\sqrt{\left[\sum_{i=1}^{N} (\psi_i{}^r)^2\right]}}{\sqrt{\left[\sum_{i=1}^{N} (f_i)^2\right]}} \times 100 \leqslant \text{TOLER}, \tag{3.27}$$

where N is the total number of nodal points in the problem and r denotes the iteration number. This criterion states that convergence occurs if the norm of the residual forces becomes less than TOLER times the norm of the total applied forces. Again the parameter NCHEK is used to indicate whether or not convergence has occurred. Three values of NCHEK are utilised:

NCHEK $= 0$	Solution has converged.
$= 1$	Solution converging, with the norm of the residual forces being less for the r^{th} iteration than the $(r-1)^{\text{th}}$ iteration.
$= 999$	Solution diverging. The norm of the residual forces is greater for the r^{th} iteration than the $(r-1)^{\text{th}}$ iteration.

Subroutine CONUND is now listed and descriptive notes provided.

```
      SUBROUTINE  CONUND                                              COND   1
C**********************************************************************COND   2
C                                                                     COND   3
C *** CHECKS FOR SOLUTION CONVERGENCE                                 COND   4
C                                                                     COND   5
C**********************************************************************COND   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE.IINCS.IITER,   COND   7
     .              KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,  COND   8
     .              NITER,NOUTP,FACTO,PVALU                           COND   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),        COND  10
     .              FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),    COND  11
     .              MATNO(25),STRES(25,2),PLAST(25),XDISP(52),        COND  12
     .              TDISP(26.2),TREAC(26.2),ASTIF(52,52),ASLOD(52),   COND  13
     .              REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)        COND  14
      DIMENSION STFOR(52),TOFOR(52)                                   COND  15
      NCHEK=0                                                         COND  16
      RESID=0.0                                                       COND  17
      RETOT=0.0                                                       COND  18
      DO 10 ISVAB=1,NSVAB                                             COND  19
      STFOR(ISVAB)=0.0                                                COND  20
   10 TOFOR(ISVAB)=0.0                                                COND  21
      DO 20 IELEM=1,NELEM                                             COND  22
      IEVAB=0                                                         COND  23
      DO 20 INODE=1,NNODE                                             COND  24
      NODNO=LNODS(IELEM,INODE)                                        COND  25
      DO 20 IDOFN=1,NDOFN                                             COND  26
      IEVAB=IEVAB+1                                                   COND  27
      NPOSN=(NODNO-1)*NDOFN+IDOFN                                     COND  28
      STFOR(NPOSN)=STFOR(NPOSN)+ELOAD(IELEM,IEVAB)                    COND  29
   20 TOFOR(NPOSN)=TOFOR(NPOSN)+TLOAD(IELEM,IEVAB)                    COND  30
      DO 30 ISVAB=1,NSVAB                                             COND  31
      REFOR=TOFOR(ISVAB)-STFOR(ISVAB)                                 COND  32
      RESID=RESID+REFOR*REFOR                                         COND  33
   30 RETOT=RETOT+TOFOR(ISVAB)*TOFOR(ISVAB)                           COND  34
      DO 40 IELEM=1,NELEM                                             COND  35
      DO 40 IEVAB=1,NEVAB                                             COND  36
   40 ELOAD(IELEM,IEVAB)=TLOAD(IELEM,IEVAB)-ELOAD(IELEM,IEVAB)        COND  37
      RATIO=100.0*SQRT(RESID/RETOT)                                   COND  38
      IF(RATIO.GT.TOLER) NCHEK=1                                      COND  39
      IF(IITER.EQ.1) GO TO 50                                         COND  40
      IF(RATIO.GT.PVALU) NCHEK=999                                    COND  41
   50 PVALU=RATIO                                                     COND  42
      WRITE(6,900) IITER,NCHEK,RATIO                                  COND  43
  900 FORMAT(1H0,5X,'ITERATION NUMBER =',I5/                          COND  44
     .       1H0,5X,'CONVERGENCE CODE =',I4,3X,                       COND  45
     .       'NORM OF RESIDUAL SUM RATIO =',E14.6)                    COND  46
      RETURN                                                          COND  47
      END                                                             COND  48
```

COND 16	Initialise the convergence indicator to zero. If convergence has not occurred during this iteration this value will be reset later in the subroutine.
COND 17	Initialise to zero the norm of the residual forces.
COND 18	Initialise to zero the norm of the total applied loads.
COND 19–21	Initialise the arrays which will contain the equivalent nodal forces and the applied loads for each nodal point.

COND 22–30 Assemble the equivalent nodal forces and applied load contributions of each *element* to give the total *nodal* values, as required for use in (3.27). This manipulation is necessary as we have decided to associate loads with an element rather than nodal points.

COND 32 Calculate the nodal residual force according to (2.4).

COND 33 Evaluate the norm of the residual forces.

COND 34 Evaluate the norm of the total applied forces.

COND 35–37 Calculate the residual nodal forces for each element, for application as forces for the next iteration according to (2.12).

COND 38 Compute the left-hand side of (3.27)—*the residual sum ratio.*

COND 39 If (3.27) is not satisfied reset NCHEK = 1 to indicate that convergence has not yet occurred.

COND 40–41 For second and subsequent iterations check to see if the residual sum ratio has decreased from the previous iteration. If not, set NCHEK = 999.

COND 42 Store the residual sum ratio, in order to perform the check indicated in COND 41 during the next iteration.

COND 43–46 Write the convergence code and the residual sum ratio.

3.10.4 Numerical examples

The numerical example considered in Section 3.9.3 and illustrated in Fig. 3.3, was reanalysed using the Newton–Raphson approach. The process converged to the nonlinear solution in 5 iterations compared to the 10 cycles required for the direct iteration method. The reduction in the number of iterations must, however, be balanced against the increased computing effort required for the solution of nonsymmetric equations. This remark is applicable only when advantage of the symmetric property of the equations is taken in solution as is the case in the more sophisticated equation solver described later in Chapter 6. The numerical results are practically identical to those obtained by the method of direct iteration and consequently both solutions are represented by the full circles in Fig. 3.3. The problem of Fig. 3.4 was also reanalysed and a similar improvement in convergence behaviour was obtained with only 7 iterations being required in place of the 12 necessitated by direct iteration.

3.11 Program for the solution of nonlinear elastic problems

In this section a program is developed which permits the solution of nonlinear elastic problems by either the tangential stiffness or the initial stiffness approach or by a combination of both methods. The options open are controlled by the parameter NALGO, the possible values of which are described in Section 3.2.

The structure of this program is identical to that described in Section 3.10 and it is only necessary to develop appropriate subroutines for element stiffness formulation, STIFF2, and residual force evaluation, REFOR2.

3.11.1 Element stiffness subroutine, STIFF2

For any value of the total strain, ϵ, in an element, the tangential stiffness matrix is explicitly given by (2.33). It is seen from this expression that the first derivative of the strain function must be known. For the calculation of the residual forces, the strain function itself must be input. Since the computer cannot perform even the simplest differentiation it is necessary to supply both quantities in the form of FUNCTION statements. As an example, the strain function will be assumed to be of the form

$$g(\epsilon) = \epsilon - 5\epsilon^2, \tag{3.28}$$

in which case

$$g'(\epsilon) = 1 - 10\epsilon. \tag{3.29}$$

Subroutine STIFF2 is now listed below.

```
      SUBROUTINE  STIFF2                                                STF2
C***********************************************************************STF2
C                                                                       STF2
C *** CALCULATES ELEMENT STIFFNESS MATRICES                             STF2
C                                                                       STF2
C***********************************************************************STF2
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     STF2
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     STF2
     .             NITER,NOUTP,FACTO,PVALU                              STF2
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          STF2
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       STF2
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           STF2
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      STF2
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           STF2
      REWIND 1                                                          STF2
      DO 10 IELEM=1,NELEM                                               STF2
      LPROP=MATNO(IELEM)                                                STF2
      YOUNG=PROPS(LPROP,1)                                              STF2
      XAREA=PROPS(LPROP,2)                                              STF2
      NODE1=LNODS(IELEM,1)                                              STF2
      NODE2=LNODS(IELEM,2)                                              STF2
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))                              STF2
      PTRAN=PLAST(IELEM)                                                STF2
      COEFF=YOUNG*XAREA/ELENG                                           STF2
      FMULT=COEFF*STDIV(PTRAN)                                          STF2
      ESTIF(1,1)=FMULT                                                  STF2
      ESTIF(1,2)=-FMULT                                                 STF2
      ESTIF(2,1)=-FMULT                                                 STF2
      ESTIF(2,2)=FMULT                                                  STF2
      WRITE(1) ESTIF                                                    STF2
   10 CONTINUE                                                          STF2
      RETURN                                                            STF2
      END                                                               STF2
```

STF2 15	Rewind the file on which the stiffness matrix of each element will be stored.
STF2 16	Loop over each element.
STF2 17	Identify the material property of each element.
STF2 18	Set YOUNG equal to the reference value of the material modulus, E_0.
STF2 19	Set XAREA equal to the cross-sectional area.
STF2 20–21	Identify the node numbers of the element.
STF2 22	Calculate the element length.
STF2 23	Set PTRAN equal to the total strain, ϵ.
STF2 24–25	Compute the multiplying term in (2.33) with $g'(\epsilon)$ given by STDIV (PTRAN).
STF2 26–29	Compute the components of the stiffness matrix.
STF2 30	Write the element stiffness matrix on to disc file.
STF2 31	Termination of DO LOOP over each element.

For a strain derivative function as defined by (3.29), the appropriate function statement is provided below.

```
      FUNCTION STDIV(PTRAN)                                            STF2  34
C****                                                                  STF2  35
C     STRAIN DERIVATIVE FUNCTION                                       STF2  36
C****                                                                  STF2  37
      STDIV=1.0-10.0*PTRAN                                             STF2  38
      RETURN                                                           STF2  39
      END                                                              STF2  40
```

3.11.2 Residual force calculation subroutine REFOR2

The residual forces existing at the end of any iteration must be calculated according to (2.4). The first step in this calculation entails the evaluation of the equivalent nodal forces, which are the forces required to produce the total displacements existing in the element. The element strain is simply

$$\epsilon_E = \begin{cases} (\phi_2-\phi_1)/L & \text{for} \quad x_2 > x_1 \\ (\phi_1-\phi_2)/L & \text{for} \quad x_2 < x_1, \end{cases} \tag{3.30}$$

where x_1 and x_2 denote the coordinates of the element nodes. This notation is required to ensure that tensile strains are positive and enables the nodal connections to be assigned in any order.

Then from (2.30) the stress in the element is given by

$$\sigma_E = E_0 g(\epsilon_E), \tag{3.31}$$

and the equivalent nodal forces are

$$f_1 = -f_2 = \begin{cases} -\sigma_E A & \text{for} \quad x_2 > x_1 \\ \sigma_E A & \text{for} \quad x_2 < x_1. \end{cases} \tag{3.32}$$

Subroutine REFOR2 is now listed and described.

```
      SUBROUTINE REFOR2                                                 RFR2   1
C***********************************************************************RFR2   2
C                                                                       RFR2   3
C *** CALCULATES INTERNAL EQUIVALENT NODAL FORCES                       RFR2   4
C                                                                       RFR2   5
C***********************************************************************RFR2   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     RFR2   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     RFR2   8
     .             NITER,NOUTP,FACTO,PVALU                              RFR2   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          RFR2  10
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       RFR2  11
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           RFR2  12
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      RFR2  13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)           RFR2  14
      DO 10 IELEM=1,NELEM                                               RFR2  15
      DO 10 IEVAB=1,NEVAB                                               RFR2  16
   10 ELOAD(IELEM,IEVAB)=0.0                                            RFR2  17
      DO 30 IELEM=1,NELEM                                               RFR2  18
      LPROP=MATNO(IELEM)                                                RFR2  19
      YOUNG=PROPS(LPROP,1)                                              RFR2  20
      XAREA=PROPS(LPROP,2)                                              RFR2  21
      NODE1=LNODS(IELEM,1)                                              RFR2  22
      NODE2=LNODS(IELEM,2)                                              RFR2  23
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))                              RFR2  24
      IF(COORD(NODE2).GT.COORD(NODE1)) STRAN=(XDISP(NODE2)-XDISP(NODE1))RFR2  25
     . /ELENG                                                           RFR2  26
      IF(COORD(NODE2).LT.COORD(NODE1)) STRAN=(XDISP(NODE1)-XDISP(NODE2))RFR2  27
     . /ELENG                                                           RFR2  28
      PLAST(IELEM)=PLAST(IELEM)+STRAN                                   RFR2  29
      PTRAN=PLAST(IELEM)                                                RFR2  30
      STRES(IELEM,1)=YOUNG*STNFN(PTRAN)                                 RFR2  31
      IF(COORD(NODE2).GT.COORD(NODE1)) GO TO 20                         RFR2  32
      ELOAD(IELEM,1)=STRES(IELEM,1)*XAREA                               RFR2  33
      ELOAD(IELEM,2)=-STRES(IELEM,1)*XAREA                              RFR2  34
      GO TO 30                                                          RFR2  35
   20 ELOAD(IELEM,1)=-STRES(IELEM,1)*XAREA                              RFR2  36
      ELOAD(IELEM,2)=STRES(IELEM,1)*XAREA                               RFR2  37
   30 CONTINUE                                                          RFR2  38
      RETURN                                                            RFR2  39
      END                                                               RFR2  40
```

RFR2 15–17 Initialise to zero the array in which the equivalent nodal forces for each element will be stored.

RFR2 18 Loop over each element.

RFR2 19 Identify the material property of each element.

RFR2 20 Set YOUNG equal to the reference value of the material modulus, E_0.

RFR2 21 Set XAREA equal to the cross-sectional area.

RFR2 22–23 Identify the node numbers of the element.

RFR2 24 Calculate the element length.

RFR2 25–28 Calculate the increase in element strain which occurred during the current iteration according to (3.30) (since XDISP measures the displacement change only).

RFR2 29 Compute the total strain.

RFR2 30–31 Compute the element stress according to (3.31).

RFR2 32–37 Compute the equivalent nodal forces according to (3.32).

RFR2 38 Termination of DO LOOP over the elements.

For calculation of the element stress in steps RFR2 30–31 (equation (3.31)) the strain function $g(\epsilon)$ must be defined. The FUNCTION statement appropriate to the variation indicated in (3.28) is provided below.

```
      FUNCTION STNFN(PTRAN)                                    RFR2  41
C****                                                          RFR2  42
C     STRAIN FUNCTION                                          RFR2  43
C****                                                          RFR2  44
      STNFN=PTRAN-5.0*PTRAN*PTRAN                              RFR2  45
      RETURN                                                   RFR2  46
      END                                                      RFR2  47
```

The equivalent nodal forces evaluated here are converted into residual forces ψ in subroutine CONUND as described in Section 3.10.3.

3.11.3 Numerical examples

The first example considered is the uniaxial loading of a two-element system. The stress/strain relationship is assumed to be defined in terms of the nonlinear expression (3.28). The applied load is incrementally increased and the combined tangential/initial stiffness solution algorithm, NALGO = 4, is employed. Figure 3.5 shows the solution behaviour during iteration to the nonlinear solution. The element stiffnesses are initially assembled at the beginning of a load increment and then kept constant during iteration to the nonlinear solution. The convergence path is plotted and it is seen that the process converges within 7 iterations for the first load increment. For the second load increment the process requires 9 iterations before convergence takes place. The process diverged rapidly on further increase of load to a total value of 11; which is expected since no solution can exist for this load value.

As an illustration of the application of the initial stiffness method to strain-softening problems, the above problem was reanalysed with the structure being loaded by prescribing an increasing value of displacement to node 3, rather than incrementing an applied load. For strain values at and beyond the peak load, the structural stiffness is either zero or negative and an initial stiffness approach must be employed. Figure 3.6 shows the results when the structure is strained beyond the peak load value.

3.12 Program for the solution of elasto-plastic problems

A computer program is now developed for the solution of one-dimensional elasto-plastic problems. Once again a tangential stiffness, initial stiffness or combined approach is permitted for solution. The program differs only from that described in the previous section in the explicit form of the element stiffness and residual force subroutines.

3.12.1 Element stiffness subroutine, STIFF3

Before yielding, the stiffness matrix of an element with linear displacement variation is given by (2.38). After the onset of plastic deformation, as

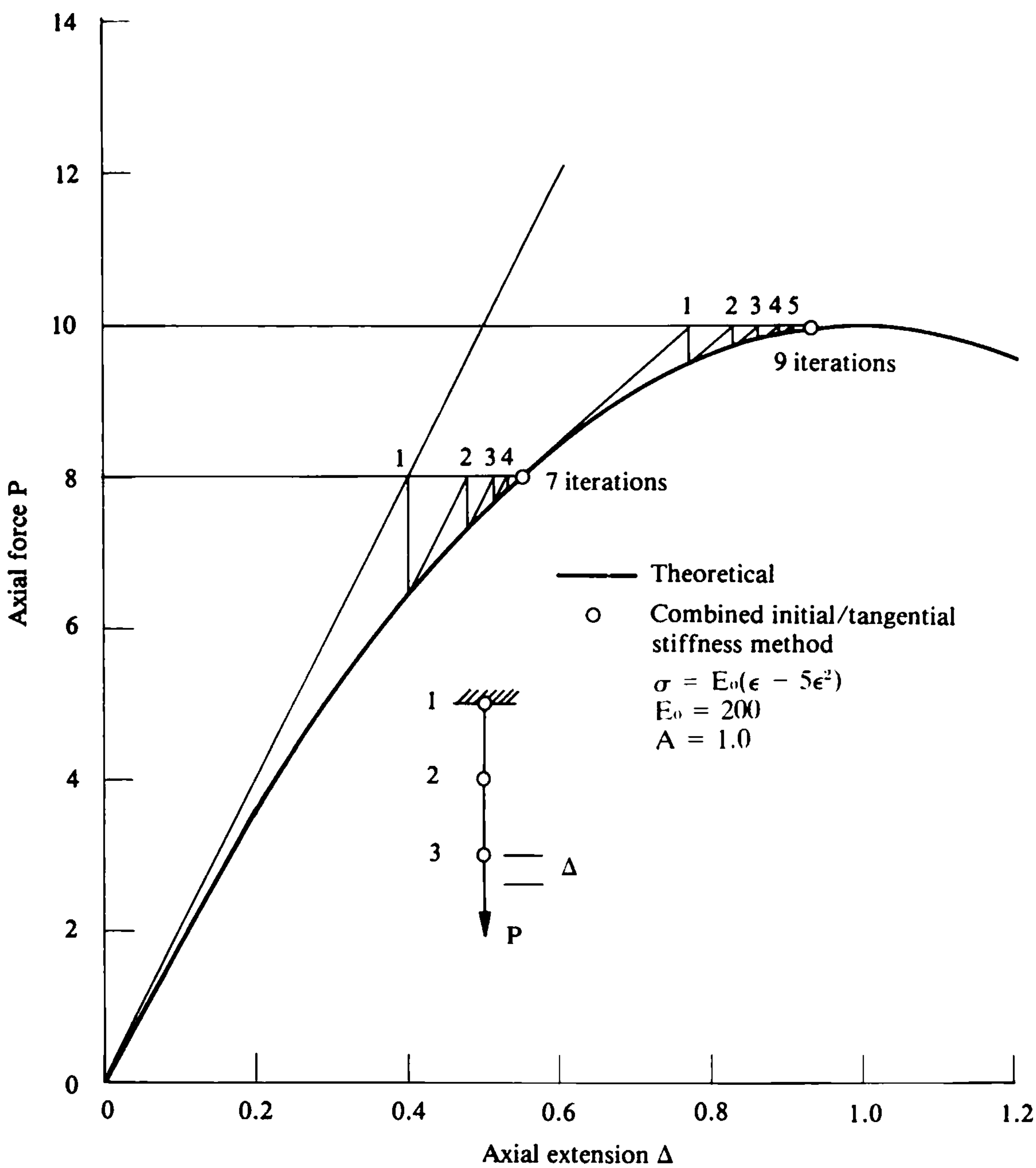


Fig. 3.5 Load/extension response of a nonlinear elastic bar under applied axial loading.

governed by the uniaxial yield stress σ_Y, the material stiffness is reduced and the elasto-plastic stiffness matrix is explicitly given by (2.43). Thus when forming the stiffness matrix for each element, it is first necessary to check whether the element behaviour is elastic or elasto-plastic. This can best be monitored by recording the plastic strain component, ϵ_p, for each element and noting that this will be zero for a completely elastic material response.

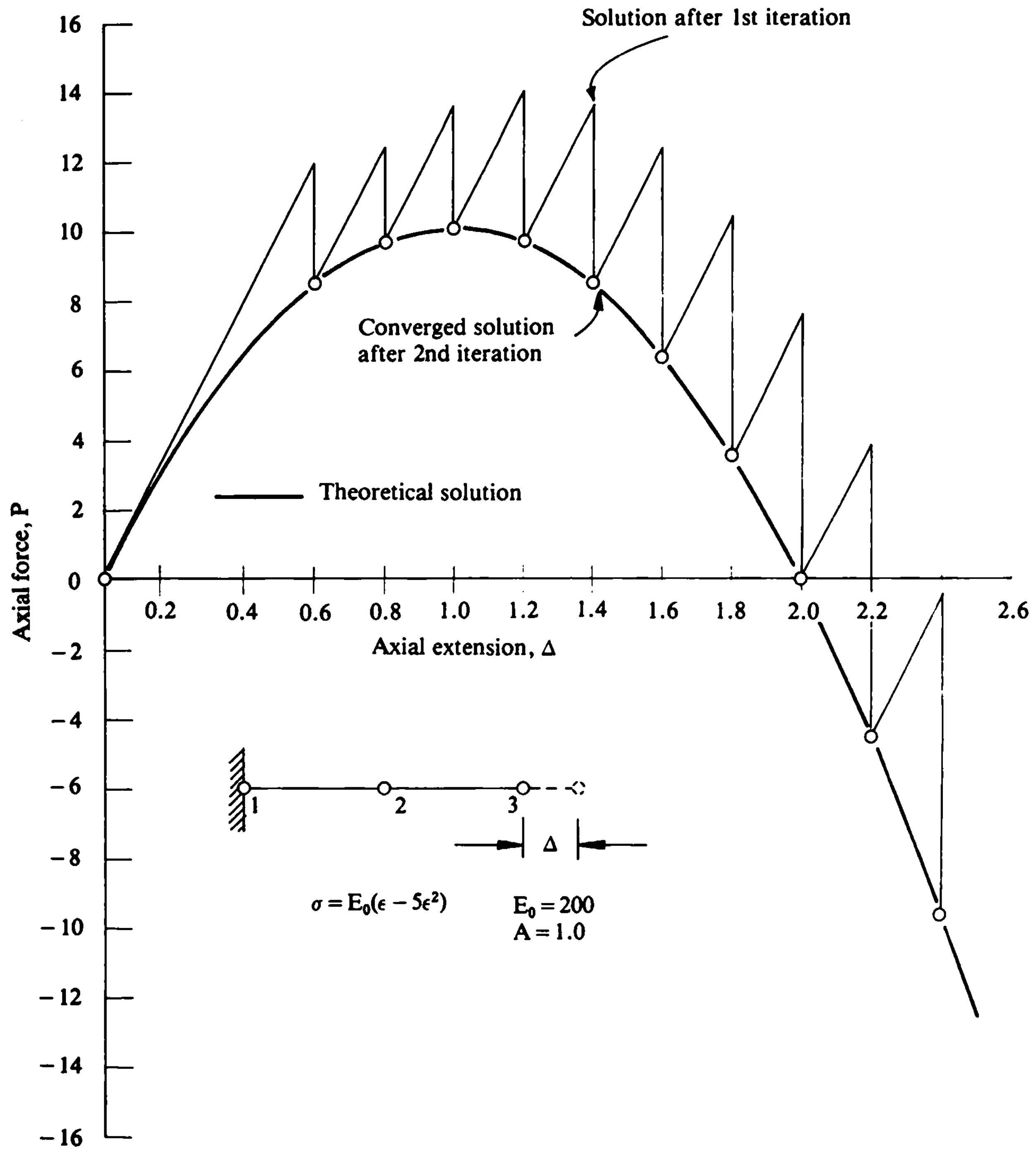


Fig. 3.6 Solution for a nonlinear elastic bar by initial stiffness, incremented prescribed displacement approach.

Subroutine STIFF3 can now be presented.

```
      SUBROUTINE  STIFF3                                                STF3
C***********************************************************************STF3
C                                                                       STF3
C *** CALCULATES ELEMENT STIFFNESS MATRICES                             STF3
C                                                                       STF3
C***********************************************************************STF3
```

```
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,    STF3   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,    STF3   8
     .             NITER,NOUTP,FACTO,PVALU                             STF3   9
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),         STF3  10
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),      STF3  11
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),          STF3  12
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),     STF3  13
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)          STF3  14
      REWIND 1                                                         STF3  15
      DO 10 IELEM=1,NELEM                                              STF3  16
      LPROP=MATNO(IELEM)                                               STF3  17
      YOUNG=PROPS(LPROP,1)                                             STF3  18
      XAREA=PROPS(LPROP,2)                                             STF3  19
      HARDS=PROPS(LPROP,4)                                             STF3  20
      NODE1=LNODS(IELEM,1)                                             STF3  21
      NODE2=LNODS(IELEM,2)                                             STF3  22
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))                             STF3  23
      FMULT=YOUNG*XAREA/ELENG                                          STF3  24
      IF(PLAST(IELEM).GT.0.0) FMULT=FMULT*(1.0-YOUNG/(YOUNG+HARDS))    STF3  25
      ESTIF(1,1)=FMULT                                                 STF3  26
      ESTIF(1,2)=-FMULT                                                STF3  27
      ESTIF(2,1)=-FMULT                                                STF3  28
      ESTIF(2,2)=FMULT                                                 STF3  29
      WRITE(1) ESTIF                                                   STF3  30
   10 CONTINUE                                                         STF3  31
      RETURN                                                           STF3  32
      END                                                              STF3  33
```

STF3 15	Rewind the file on which the stiffness matrix of each element will be stored.
STF3 16	Loop over each element.
STF3 17	Identify the material property of each element.
STF3 18	Set YOUNG equal to the material elastic modulus.
STF3 19	Set XAREA equal to the cross-sectional area.
STF3 20	Set HARDS equal to the strain hardening parameter, H'.
STF3 21–22	Identify the node numbers of the element.
STF3 23	Calculate the element length.
STF3 24	Compute the multiplying term in (2.38) as FMULT.
STF3 25	Check if the element has yielded. If yes, compute FMULT as the multiplying term in (2.43).
STF3 26–29	Compute the components of the stiffness matrix.
STF3 30	Write the element stiffness matrix on to disc file.
STF3 31	Termination of DO LOOP over each element.

3.12.2 Residual force subroutine, REFOR3

The purpose of this subroutine is to calculate the equivalent nodal forces from which the residual nodal forces will be evaluated in subroutine CONUND. In view of the essentially incremental nature of the equations of plasticity, the subroutine is somewhat more intricate than the residual force subroutines developed to date. All stress and strain components must be accumulated from the values obtained during each iteration. The situation is further complicated by the fact that an element may yield when the residual forces are applied as loads for any iteration. The precise load at which yielding begins will generally lie somewhere between the total load corresponding to the previous iteration and the total load for the present cycle. Consequently the yield load must be determined and the plastic strain computed for only the post yield portion of the load. The general procedure adopted is to determine the stress in each element so that the yield criterion is satisfied. If the actual stress in any element is greater than this permissible value, then the additional part is removed but is included in the residual force vector to maintain equilibrium.

Consider the situation existing for the r^{th} iteration of any particular load increment. The solution algorithm employed is presented below.

Step a The applied loads for the r^{th} iteration are the residual forces ψ^{r-1} calculated at the end of the $(r-1)^{\text{th}}$ iteration according to (2.4). These applied loads give rise to displacement increments, $\Delta\varphi^r$, according to (2.12). Hence calculate the corresponding increment of strain $\Delta\epsilon^r$. For the general element denote this value by $\Delta\epsilon^r$ and it is shown in Fig. 3.7.

Step b Compute the incremental stress change assuming linear elastic behaviour. This will introduce errors if the element has yielded and the material is behaving elasto-plastically. However, we will correct any discrepancy when the residual forces are calculated. Therefore we calculate the stress change according to $\Delta\sigma_e^r = E\Delta\epsilon^r$, where the subscript e is used to denote that this stress is based on elastic behaviour.

Step c Accumulate the total stress for each element as $\sigma_e^r = \sigma^{r-1}+\Delta\sigma_e^r$. The stress σ^{r-1} will have been determined to satisfy the yield condition during the $(r-1)^{\text{th}}$ iteration. Consequently, the error in the stress σ_e^r is limited to $\Delta\sigma_e^r$. Again the subscript e denotes that σ_e^r is based on an elastic behaviour.

Step d The next step in the process depends on whether or not the element had previously yielded on the $(r-1)^{\text{th}}$ iteration. This can be checked from the known value of the yield stress for the $(r-1)^{\text{th}}$ iteration. The stress limit for this cycle is given from Fig. 2.9 as

$$\sigma_Y{}^{r-1} = \sigma_Y + H'\epsilon_p{}^{r-1}.$$

Since the plastic strain ϵ_p will differ from element to element, each element will generally have a different permissible stress level.

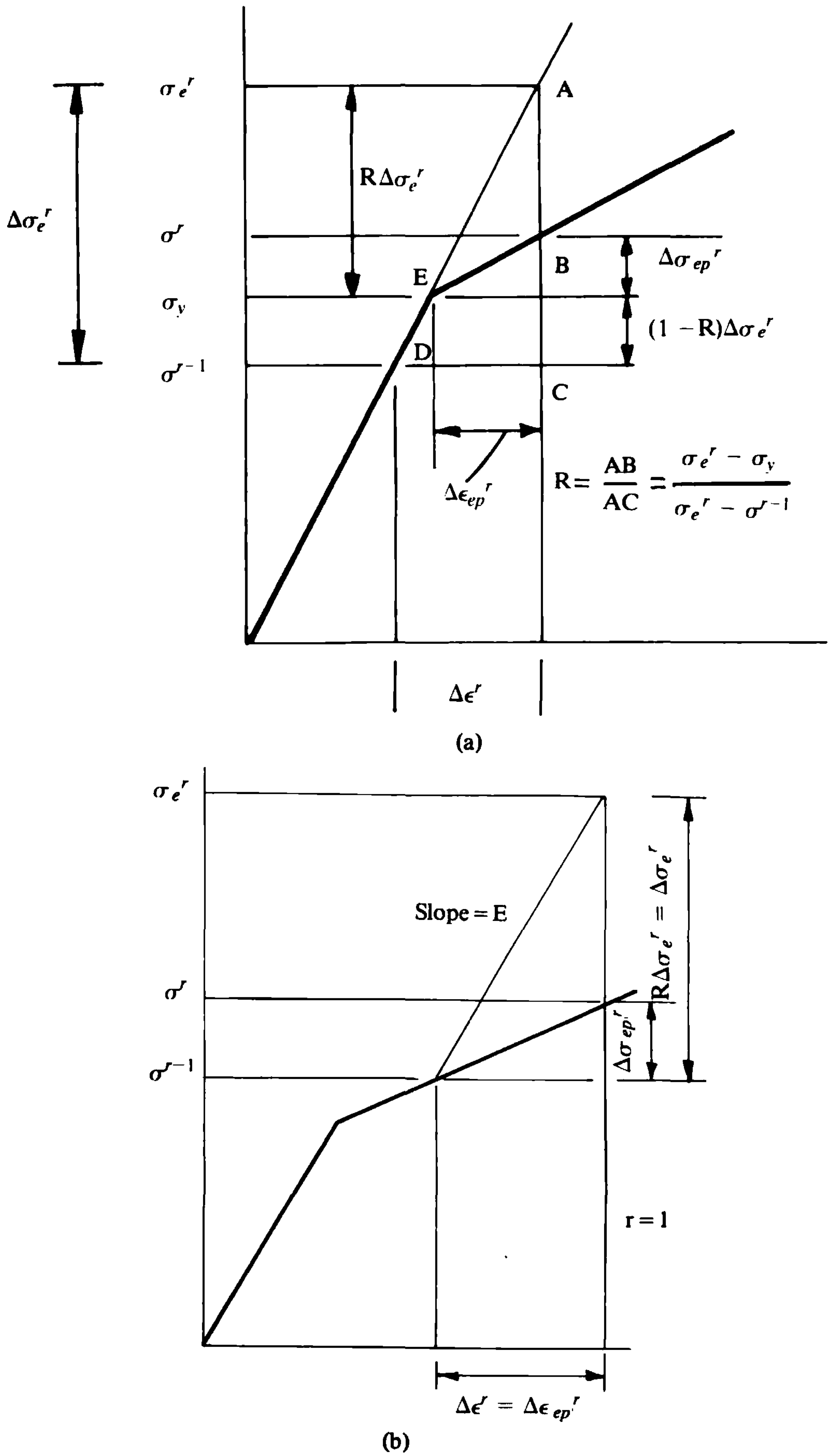


Fig. 3.7 Incremental stress and strain changes in a one-dimensional elasto-plastic material. (a) Initial yielding of material. (b) Material previously yielded.

Therefore we check if $\sigma^{r-1} > \sigma_Y + H'\epsilon_p{}^{r-1}$. If the answer is:

YES

which implies that the element had already yielded during the previous iteration, then check to see if $\sigma_e{}^r > \sigma^{r-1}$. If the answer is:

NO	YES
The element is unloading which according to plasticity theory must take place elastically, and no further action need be taken. Go directly to Step g.	The element had reached the threshold stress during the previous iteration and the stress is still increasing. Therefore all the excess stress $\sigma_e{}^r - \sigma^{r-1}$ must be reduced to the yield value as indicated in Fig. 3.7(b). Therefore the factor, R, which defines the portion of the stress which must be modified to satisfy the yield condition, is equal to 1 in this case as shown in Fig. 3.7(b).

NO

which implies that the element had not previously yielded. We now check to see if $\sigma_e{}^r > \sigma_Y$. If the answer is:

NO	YES
The element is still elastic and no further action need be taken. Go directly to Step g.	The element has yielded during the application of load corresponding to this iteration as illustrated in Fig. 3.7(a). Therefore the portion of the stress greater than the yield value must be reduced to the elasto-plastic line. The removed portion will be included in the residual force vector. The reduction factor, R, is found, with reference to Fig. 3.7(a) to be $R = \dfrac{AB}{AC} = \dfrac{\sigma_e{}^r - \sigma_Y}{\sigma_e{}^r - \sigma^{r-1}}.$

Step e *For yielded elements only*, calculate the increment of stress $\Delta\sigma_{ep}{}^r$, which is the portion after yielding, permitted by elasto-plastic theory. This stress value is shown in Fig. 3.7 for the two cases when (a) yielding has commenced during this iteration and (b) when the element had previously yielded. Using (2.4) we have

$$\Delta\sigma_{ep}{}^r = E\left(1 - \frac{E}{E+H'}\right)\Delta\epsilon_{ep}{}^r, \tag{3.33}$$

where the subscript *ep* denotes elasto-plastic behaviour. For the above to be generally true we must restrict ourselves to small increments of stress and strain. For the situation of Fig. 3.7(a), noting that triangles ADC and AEB are similar, we have

$$\Delta\epsilon_{ep}{}^r = R\Delta\epsilon^r. \tag{3.34}$$

Defining $R = 1$ for the situation of Fig. 3.7(b), then (3.34) is still correct. Therefore

$$\Delta\sigma_{ep}{}^r = E\left(1 - \frac{E}{E+H'}\right)R\Delta\epsilon^r. \tag{3.35}$$

The total current stress is given by

$$\sigma^r = \sigma^{r-1} + (1-R)\Delta\sigma_e{}^r + \Delta\sigma_{ep}{}^r, \tag{3.36}$$

where the second term accounts for the elastic portion of the stress increment occurring before the onset of yielding.

Step f *For yielded elements only*, evaluate the total plastic strain for the element as $\epsilon_p{}^r = \epsilon_p{}^{r-1} + \Delta\epsilon_p{}^r$ where the plastic strain increment for the iteration is calculated as follows. For the elastic component of strain, $\Delta\epsilon_e{}^r$, we have

$$\Delta\epsilon_e{}^r = \frac{\Delta\sigma^r}{E}. \tag{3.37}$$

Substituting for $\Delta\sigma^r$ from the linearised form of (2.35) into (3.37) and then using (2.34) we obtain

$$\Delta\epsilon_p{}^r = \frac{\Delta\epsilon^r}{1+H'/E}. \tag{3.38}$$

Since the plastic strain component must be calculated for the part of the strain after the element yields, then, with reference to Fig. 3.7, $\Delta\epsilon^r$ must be replaced by $\Delta\epsilon_{ep}{}^r$. Or, using (3.34), we have

$$\Delta\epsilon_p{}^r = \frac{R\Delta\epsilon^r}{1+H'/E}. \tag{3.39}$$

Then the total current plastic strain for the element is

$$\epsilon_p{}^r = \epsilon_p{}^{r-1} + \frac{R\Delta\epsilon^r}{1+H'/E}. \tag{3.40}$$

Step g *For elastic elements only*, store the correct current stress as

$$\sigma^r = \sigma^{r-1} + \Delta\sigma_e{}^r. \tag{3.41}$$

(This in fact repeats Step c.)

Step h Finally, calculate the equivalent nodal forces from the element stress according to

$$f_1 = -f_2 = \begin{cases} -\sigma^r A & \text{for} \quad x_2 > x_1 \\ \sigma^r A & \text{for} \quad x_2 < x_1. \end{cases} \tag{3.42}$$

Subroutine REFOR3 is now presented below and explanatory notes provided.

```
      SUBROUTINE REFOR3
C*********************************************************************
C
C *** CALCULATES INTERNAL EQUIVALENT NODAL FORCES
C
C*********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO,PVALU
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=0.0
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      YOUNG=PROPS(LPROP,1)
      XAREA=PROPS(LPROP,2)
      YIELD=PROPS(LPROP,3)
      HARDS=PROPS(LPROP,4)
      NODE1=LNODS(IELEM,1)
      NODE2=LNODS(IELEM,2)
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))
      IF(COORD(NODE2).GT.COORD(NODE1)) STRAN=(XDISP(NODE2)-XDISP(NODE1))
     . /ELENG
      IF(COORD(NODE2).LT.COORD(NODE1)) STRAN=(XDISP(NODE1)-XDISP(NODE2))
     . /ELENG
      STLIN=YOUNG*STRAN
      STCUR=STRES(IELEM,1)+STLIN
      PREYS=YIELD+HARDS*ABS(PLAST(IELEM))
      IF(ABS(STRES(IELEM,1)).GE.PREYS) GO TO 20
      ESCUR=ABS(STCUR)-PREYS
      IF(ESCUR.LE.0.0) GO TO 40
      RFACT=ESCUR/ABS(STLIN)
      GO TO 30
   20 IF(STRES(IELEM,1).GT.0.0.AND.STLIN.LE.0.0) GO TO 40
      IF(STRES(IELEM,1).LT.0.0.AND.STLIN.GT.0.0) GO TO 40
      RFACT=1.0
   30 REDUC=1.0-RFACT
      STRES(IELEM,1)=STRES(IELEM,1)+REDUC*STLIN+RFACT*YOUNG*(1.0-
     . YOUNG/(YOUNG+HARDS))*STRAN
      PLAST(IELEM)=PLAST(IELEM)+RFACT*STRAN*YOUNG/(YOUNG+HARDS)
      GO TO 50
   40 STRES(IELEM,1)=STRES(IELEM,1)+STLIN
   50 IF(COORD(NODE2).GT.COORD(NODE1)) GO TO 60
      ELOAD(IELEM,1)=STRES(IELEM,1)*XAREA
      ELOAD(IELEM,2)=-STRES(IELEM,1)*XAREA
      GO TO 70
   60 ELOAD(IELEM,1)=-STRES(IELEM,1)*XAREA
      ELOAD(IELEM,2)=STRES(IELEM,1)*XAREA
   70 CONTINUE
      RETURN
      END
```

RFR3 15–17 Initialise to zero the array in which the equivalent nodal forces for each element will be stored.
RFR3 18 Loop over each element.
RFR3 19 Identify the material property of each element.
RFR3 20 Set YOUNG equal to the elastic modulus, E, of the material.
RFR3 21 Set XAREA equal to the cross-sectional area.
RFR3 22 Set YIELD equal to the uniaxial yield stress, σ_Y, of the material.
RFR3 23 Set HARDS equal to the hardening parameter, H', of the material.
RFR3 24–25 Identify the node numbers of the element.
RFR3 26 Calculate the element length.
RFR3 27–30 Calculate the element strain, so that a tensile strain is positive.
RFR3 31 Calculate $\Delta\sigma_e{}^r$ according to Step b.
RFR3 32 Calculate $\sigma_e{}^r$ according to Step c.
RFR3 33–34 Check if the element had yielded on the previous iteration, i.e., if $\sigma^{r-1} > \sigma_Y + H' \epsilon_p{}^{r-1}$ which is the first operation of Step d. The absolute value of σ^{r-1} is taken to account for yielding in compression.
RFR3 35–36 If the element was previously elastic, check to see if it has yielded during this iteration.
RFR3 37 For an element which yields during this iteration, calculate

$$R = \frac{\sigma_e{}^r - \sigma_Y}{\sigma_e{}^r - \sigma^{r-1}}$$

(Fig. 3.7(a)). The absolute value sign is taken to account for compressive loading.
RFR3 39–40 Check to see if an element which had previously yielded is unloading during this iteration. If yes, go to 40.
RFR3 41 Otherwise, set $R = 1$.
RFR3 42 Evaluate, $(1-R)$.
RFR3 43–44 For plastic elements, calculate the correct current stress, σ^r, according to (3.36).
RFR3 45 Also calculate the plastic strain, $\epsilon_p{}^r$, according to (3.40).
RFR3 47 For elastic elements, calculate the current stress, σ^r, according to Step g.
RFR3 48–53 Evaluate the equivalent nodal forces, according to Step h.
RFR3 54 Termination of DO LOOP over the elements.

3.12.3 Numerical examples

The first example considered is the yielding of a bar under self weight loading. The problem and finite element idealisation employed is illustrated in Fig. 3.8. Progressive yielding is induced in the system by increasing the gravitational field incrementally. The gravitational force due to self weight

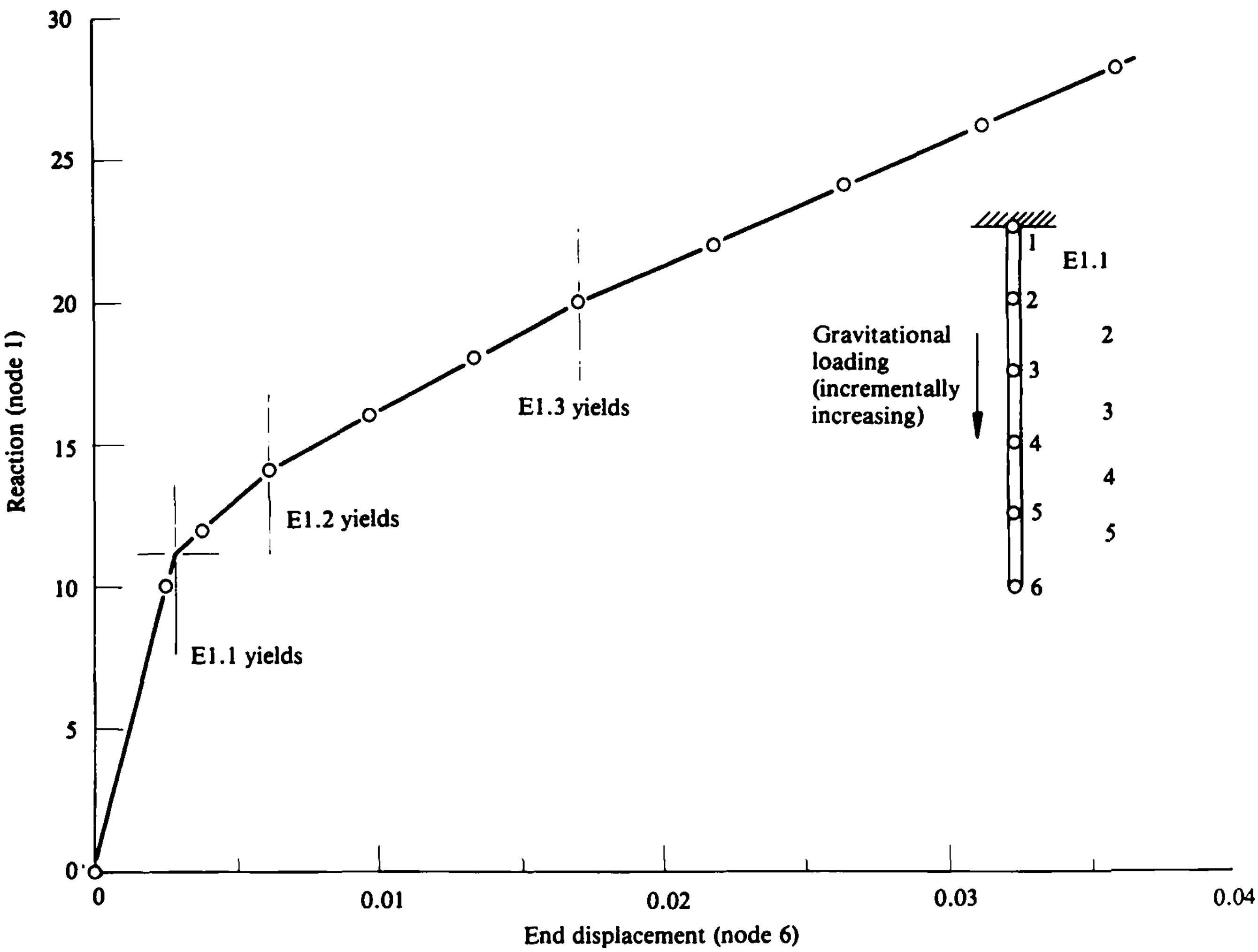


Fig. 3.8 Load/displacement response of a vertical bar loaded by a progressively increasing self-weight.

acting on each element is equally distributed to its two nodes. The structure is capable of carrying load beyond first yield, due to the strain hardening characteristic of the material.

The second example considered is the compound bar shown in Fig. 3.9. The two bars have a different yield stress and cross-sectional area in order to induce differential yielding. The structure is loaded by an end load, P, which is systematically incremented. The load/extension characteristics for the system are shown in Fig. 3.9. It is seen that there is an initial loss of stiffness corresponding to yielding of the first bar followed by a further reduction when the second bar becomes plastic.

This simple example suggests a method by which more complex material responses can be generated. By connecting two bars with different properties in parallel we obtain a material behaviour made up of three linear portions.

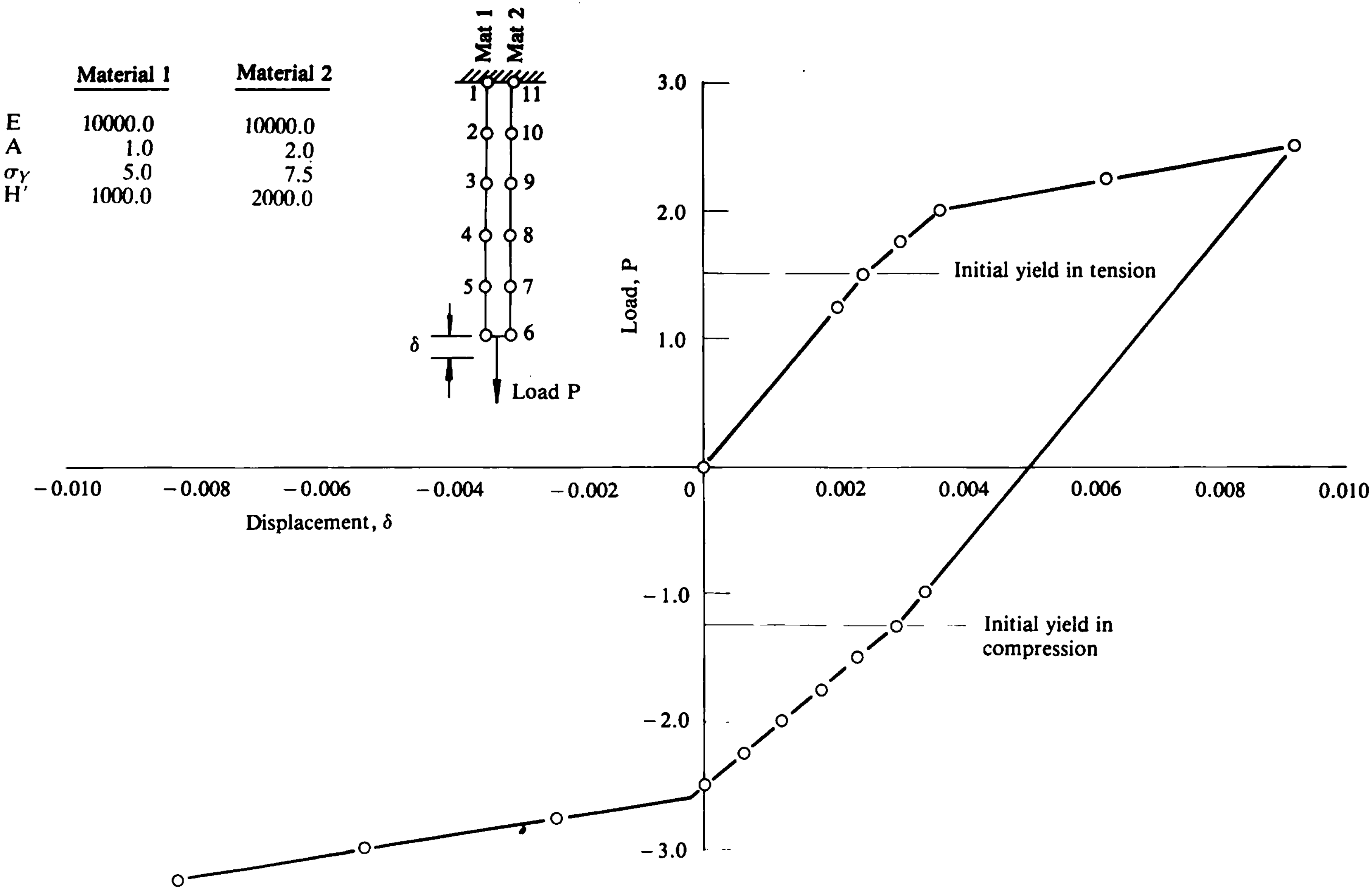


Fig. 3.9 Load cycling of an elasto-plastic parallel bar model.

By connecting n bars in parallel and choosing the yield stress and cross-sectional area of each appropriately we can approximate any arbitrary stress/strain response piecewise linearly by $(n+1)$ intervals. This is the basis of the 'overlay method'[7] which will be described later in Chapter 8.

Also included in Fig. 3.9 are the results for the case when the load is cycled. First the load is incremented in tension up to a certain level, then removed and applied compressively, before final removal. It is immediately seen that a Bauschinger effect[8] is obtained with initial yield in compression taking place at a reduced value. This occurs even though we have assumed an equal yield stress in tension and compression. This behaviour is attributable to the differential straining of the two components and is a phenomenon evident in real materials.

3.13 Problems

3.1 Reanalyse the problem of Fig. 3.3, Section 3.9.3, for the case where the term K is assumed to vary with the unknown ϕ according to

$$K = 10(1+e^{3\phi}).$$

Use the direct iteration solution code QUITER, user instructions for which are provided in Appendix I, Section A1.1, for solution.

3.2 Resolve Problem 3.1 using the Newton–Raphson procedure which is coded in program QUNEWT. User instructions for this program are provided in Appendix I, Section A1.2. Compare the computation times required for the two different solution procedures.

3.3 The quasi-harmonic equation described in Section 2.3 is also applicable to groundwater flow problems.[5] In this application ϕ is the pressure head potential, K is the material permeability and Q is the rate at which water is being injected per unit volume of material. The flow velocity at any point is then given by $v = -K(d\phi/dx)$. Figure 3.10 illustrates the problem of water seeping through two permeable strata whose permeabilities depend on the seepage velocity as shown. By treating the problem as one-dimensional in the vertical direction obtain a numerical solution for the steady state potential and velocity distribution in the two strata.

3.4 Following the approach of Section 2.3, develop the stiffness matrix $\boldsymbol{H}^{(e)}$ and the load vector $\boldsymbol{f}^{(e)}$ for the one-dimensional axisymmetric situation. In this application all quantities are symmetric with respect to a central axis and the radial coordinate r now replaces x.

3.5 Implement the formulation of Problem 3.4 in program QUITER.

3.6 Use the computer code developed in Problem 3.5 to solve the problem of water flow in the horizontal place of the confined aquifer shown in Fig. 3.11. In this case ϕ is the piezometric head, K is the material permeability and Q is the rate at which water is being injected per unit volume of material.

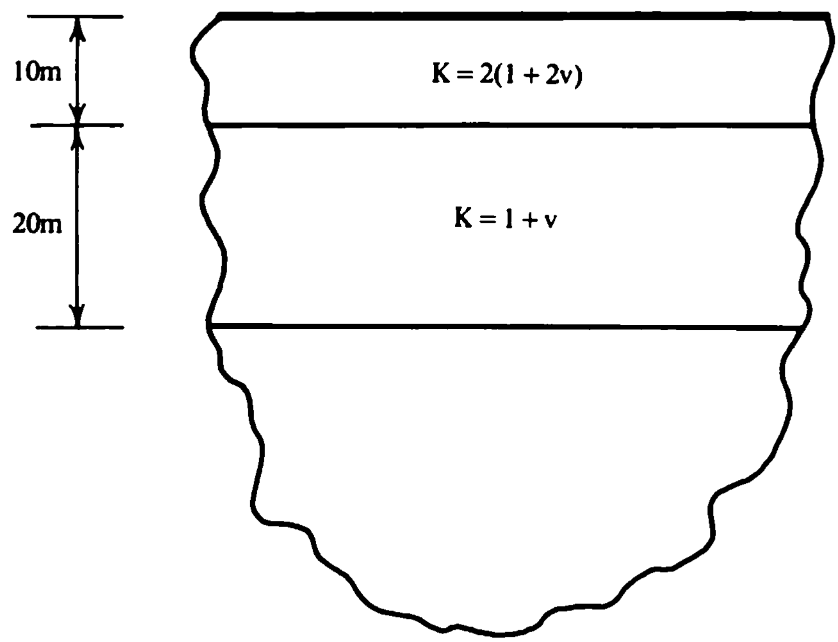


Fig. 3.10 Groundwater flow example—Problem 3.3.

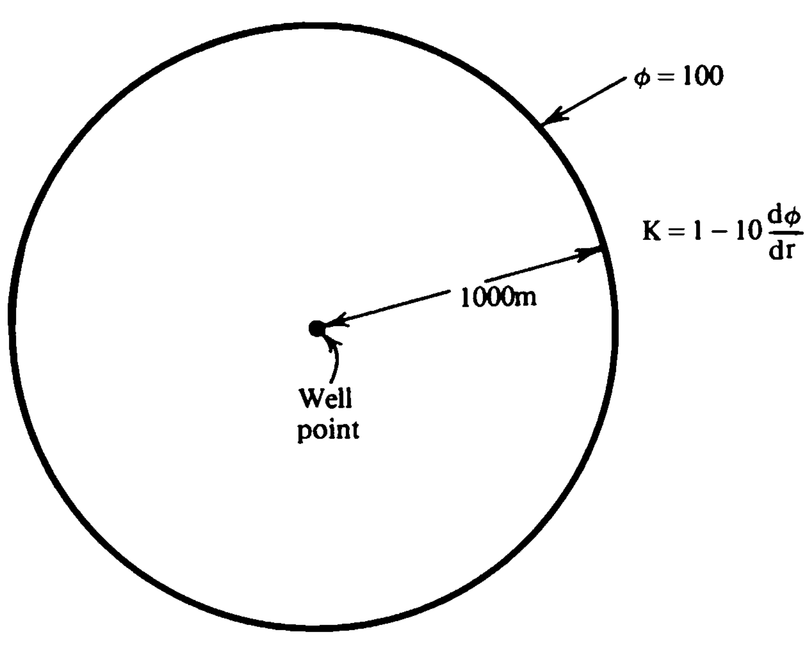


Fig. 3.11 Water flow in a confined aquifer—Problem 3.6.

The circular region shown in Fig. 3.11 has a central well point at which water is being extracted at a rate of 200 m³/day. Determine the steady state potential distribution for this system assuming the material permeability to be nonlinear in the manner shown.

3.7 The relationship between stress, σ, and strain, ϵ, for a certain locking material is given by the relationship

$$\sigma = \frac{E_0 \epsilon}{\epsilon_L(\epsilon_L - \epsilon)}, \tag{3.43}$$

in which E_0 is the elastic modulus and ϵ_L is the limiting strain value of the material. Implement this relation in program NONLAS documented in Appendix I, Section A1.3, by modifying the strain derivative function in Section 3.11.1. Also allow the behaviour of certain elements to be linear elastic. Use this modified program to determine the force displacement/relationship of the central node in Fig. 3.12 for a total applied load of 100 units.

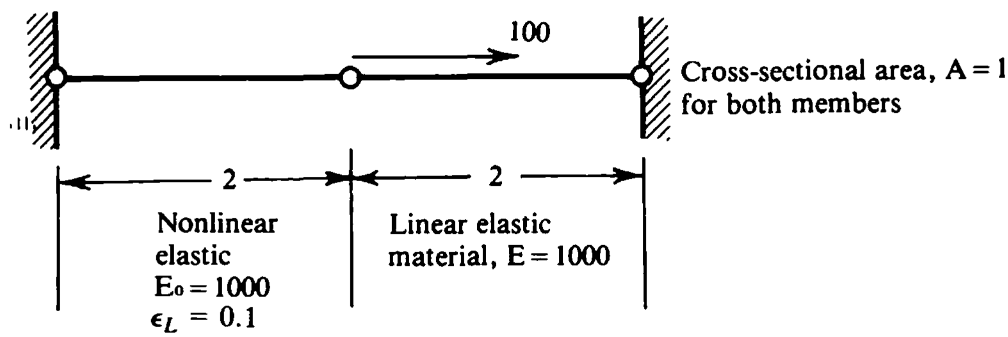


Fig. 3.12 Nonlinear elastic example—Problem 3.7.

3.8 Use program ELPLAS, for which user instructions are provided in Appendix I, Section A1.4, to solve the one-dimensional elasto-plastic problem shown in Fig. 3.13.

3.9 Develop the elastic stiffness matrix, $K^{(e)}$, for a two-node finite element in the form of a thin disc of thickness t which is to be subjected to axisymmetric in-plane loading. Assume a linear variation between nodes, as shown in Fig. 2.7, and note the following relationships

$$\epsilon_r = \frac{du}{dr} = \frac{1}{E}(\sigma_r - \nu\sigma_\theta)$$

$$\epsilon_\theta = \frac{u}{r} = \frac{1}{E}(\sigma_\theta - \nu\sigma_r), \tag{3.44}$$

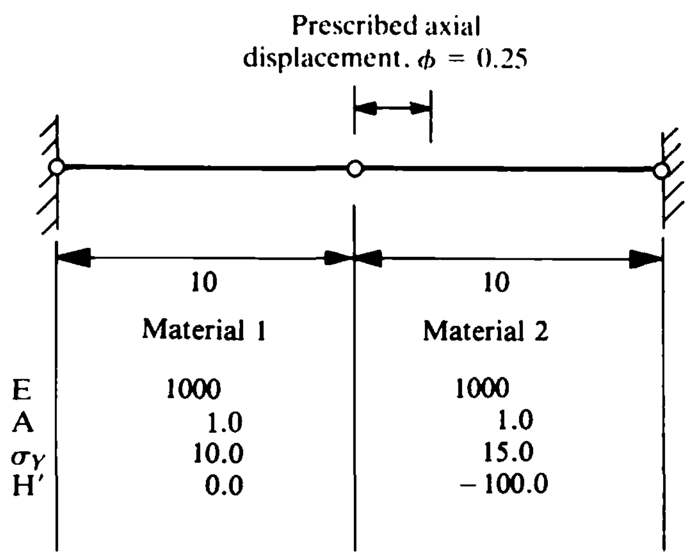


Fig. 3.13 Elasto-plastic example—Problem 3.8.

in which u is the radial displacement and E and ν are respectively the elastic modulus and Poisson's ratio of the material. Also express the stresses σ_r and σ_θ in terms of the nodal displacements ϕ_1 and ϕ_2.

3.10 Use the stiffness matrix evaluated in Problem 3.9 to modify program ELPLAS to allow solution of one-dimensional axisymmetric problems by the initial stiffness method. Assume a Tresca yield criterion (discussed in Chapter 7) where yielding is assumed to begin when the maximum shearing stress reaches a critical value. For the present application this implies commencement of yielding when either σ_r or σ_θ reaches the uniaxial yield stress, σ_Y.

3.11 Employ the program developed in Problem 3.10 to determine the elasto-plastic stress distribution in a thin disc, of thickness 1 mm, subjected to internal pressure loading. Take the internal and external

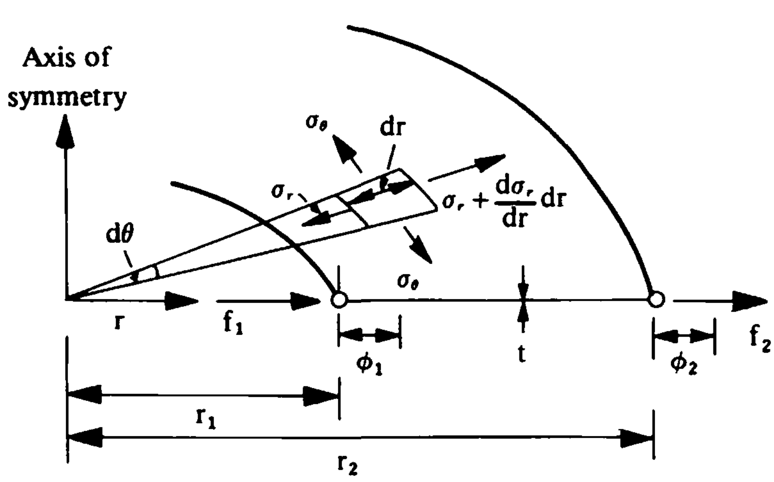


Fig. 3.14 Axisymmetric membrane element—Problem 3.9

radii of the disc as 5 cm and 10 cm respectively, the elastic modulus $E = 2 \times 10^5$ N/mm^2, Poisson's ratio $\nu = 0{\cdot}3$ and the uniaxial yield stress, $\sigma_Y = 300$ N/mm^2. Compare your solution with the theoretical expressions given in Ref. 8.

3.14 References

1. RALSTON, A., *A First Course in Numerical Analysis*, McGraw-Hill, 1965.
2. JENNINGS, A., *Matrix Computation for Engineers and Scientists*, John Wiley, 1977.
3. HINTON, E. and OWEN, D. R. J., *Finite Element Programming*, Academic Press, London, 1977.
4. SALVADORI, M. G. and BARON, M. L., *Numerical Methods in Engineering*, Prentice-Hall, New Jersey, 1964.
5. HINTON, E. and OWEN, D. R. J., *An Introduction to Finite Element Computations*, Pineridge Press, Swansea, U.K., 1979.
6. BARRER, R. M., Measurement of diffusion and thermal conductivity 'constants' in non-homogeneous media, and in media where these constants depend respectively on concentration or temperature, *Proc. Phys. Soc.* **58**, 321–331 (1946).
7. OWEN, D. R. J., PRAKASH, A. and ZIENKIEWICZ, O. C., Finite element analysis of non-linear composite materials by use of overlay systems, *Computers and Structures*, **4**, 1251–1267 (1974).
8. HILL, R., *The Mathematical Theory of Plasticity*, Oxford University Press, 1950.

Chapter 4
Viscoplastic problems in one dimension

4.1 Introduction

In this chapter the basic concepts of viscoplasticity are introduced by the consideration of one-dimensional situations. This topic is then studied further in Chapter 8 where the case of a general continuum is treated.

Viscoplastic theory allows the modelling of time rate effects in the plastic deformation process. Thus after initial yielding of the material the plastic flow, and the resulting stresses and strains, are time dependent. Such effects are always present to some degree in all materials but they may or may not be significant depending on the physical situation being considered.

The basic theory of viscoplasticity in one dimension is developed and a numerical solution process is then described. All the essential features of viscoplasticity can be demonstrated with reference to one-dimensional behaviour. Finally the solution process is coded in FORTRAN to form a working program and the basic characteristics of a viscoplastic material response are illustrated by the solution of numerical examples.

4.2 Basic theory

The concept of viscoplastic material behaviour is best introduced by means of the one-dimensional rheological model illustrated in Fig. 4.1. The friction slider component develops a stress σ_p, becoming active only if $\sigma > Y$, where σ is the total applied stress and Y is some limiting yield value. The excess stress $\sigma_d = \sigma - \sigma_p$ is carried by the viscous dashpot. Instantaneous elastic response is, of course, provided by the linear spring. The presence of the dashpot allows the stress level to instantaneously exceed the value predicted by plasticity theory, the solution tending to this equilibrium level as steady state conditions are achieved in the system.

The total strain in the model is given by the sum of the elastic and viscoplastic components as

$$\epsilon = \epsilon_e + \epsilon_{vp}. \tag{4.1}$$

The stress in the linear spring is equal to the total applied stress and is

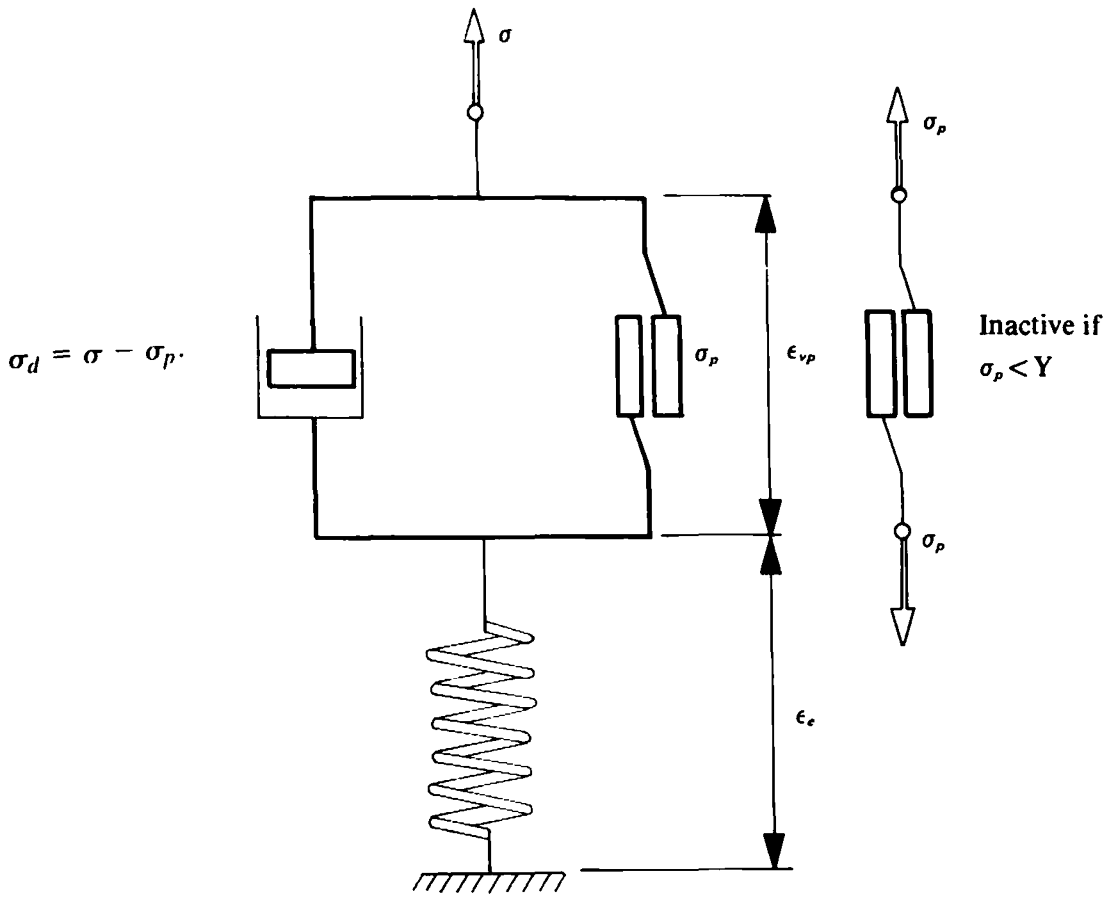


Fig. 4.1 Basic one-dimensional elastic-viscoplastic model.

related to the elastic strain by

$$\sigma_e = \sigma = E\epsilon_e, \tag{4.2}$$

where E is the elastic modulus of the linear spring.

The stress level in the friction slider depends on whether or not the threshold or yield stress, Y, has been reached. The onset of viscoplastic deformation is governed by a uniaxial yield stress σ_Y. The stress level for continuing viscoplastic flow depends on the strain-hardening characteristics of the material. Restricting discussion to a linear strain-hardening response as discussed in Section 2.5, the stress level for viscoplastic yielding at any stage is given by

$$Y = \sigma_Y + H'\epsilon_{vp}, \tag{4.3}$$

in which H' is the slope of the strain hardening portion of the stress–strain curve after removal of the elastic strain component and ϵ_{vp} is the current viscoplastic strain. Thus the stress in the friction slider is

$$\begin{aligned} \sigma_p &= \sigma \quad \text{if} \quad \left|\begin{array}{l}\sigma_p < Y \\ \sigma_p \geqslant Y.\end{array}\right. \\ &= Y \end{aligned} \tag{4.4}$$

The stress in the viscous dashpot, σ_d, is related to the viscoplastic strain by

$$\sigma_d = \mu \frac{d\epsilon_{vp}}{dt}, \tag{4.5}$$

where μ is a *viscosity coefficient* and t denotes the time. We note that

$$\sigma = \sigma_d + \sigma_p. \tag{4.6}$$

Before the onset of viscoplastic yielding $\epsilon_{vp} = 0$, giving $\sigma_d = 0$ from (4.5) and consequently $\sigma_p = \sigma$. It now remains to establish the constitutive relationship for the model under both elastic and elasto-viscoplastic conditions.

Before viscoplastic yielding, $\epsilon_{vp} = 0$ and from (4.1) and (4.2) we have the *elastic stress–strain relation* to be

$$\sigma = E\epsilon. \tag{4.7}$$

Substituting from (4.4) and (4.5) in (4.6) gives

$$\sigma_Y + H'\epsilon_{vp} + \mu \frac{d\epsilon_{vp}}{dt} = \sigma. \tag{4.8}$$

Substituting for ϵ_{vp} from (4.1) and using (4.2) results in

$$H'E\epsilon + \mu E \frac{d\epsilon}{dt} = H'\sigma + E(\sigma - \sigma_Y) + \mu \frac{d\sigma}{dt}, \tag{4.9}$$

which is a first order ordinary differential equation defining the time-dependent relationship between stress and strain under viscoplastic conditions. At this stage we introduce a *fluidity parameter*, γ, such that

$$\gamma = \frac{1}{\mu}. \tag{4.10}$$

Substituting in (4.9) and rearranging

$$\dot{\epsilon} = \frac{\dot{\sigma}}{E} + \gamma[\sigma - (\sigma_Y + H'\epsilon_{vp})], \tag{4.11}$$

in which $(\dot{\ })$ denotes derivative with respect to time, t. Or

$$\dot{\epsilon} = \dot{\epsilon}_e + \dot{\epsilon}_{vp}, \tag{4.12}$$

where

$$\dot{\epsilon}_e = \frac{\dot{\sigma}}{E}, \tag{4.13}$$

and

$$\dot{\epsilon}_{vp} = \gamma[\sigma - (\sigma_Y + H'\epsilon_{vp})]. \tag{4.14}$$

Expression (4.14) defines the viscoplastic strain rate in terms of the portion of stress in excess of the steady state yield value.

It is instructive to consider the closed form solution to (4.9). Consider the case when a constant applied stress $\sigma = \sigma_A$ is applied to the model. Then (4.9) reduces, (using (4.10)), to

$$\gamma H' \epsilon + \frac{d\epsilon}{dt} = \frac{\gamma H'}{E} \sigma_A + \gamma(\sigma_A - \sigma_Y). \tag{4.15}$$

The solution to this first-order ordinary differential equation is elementary and is

$$\epsilon = \frac{\sigma_A}{E} + \frac{(\sigma_A - \sigma_Y)}{H'}[1 - e^{-H'\gamma t}], \tag{4.16}$$

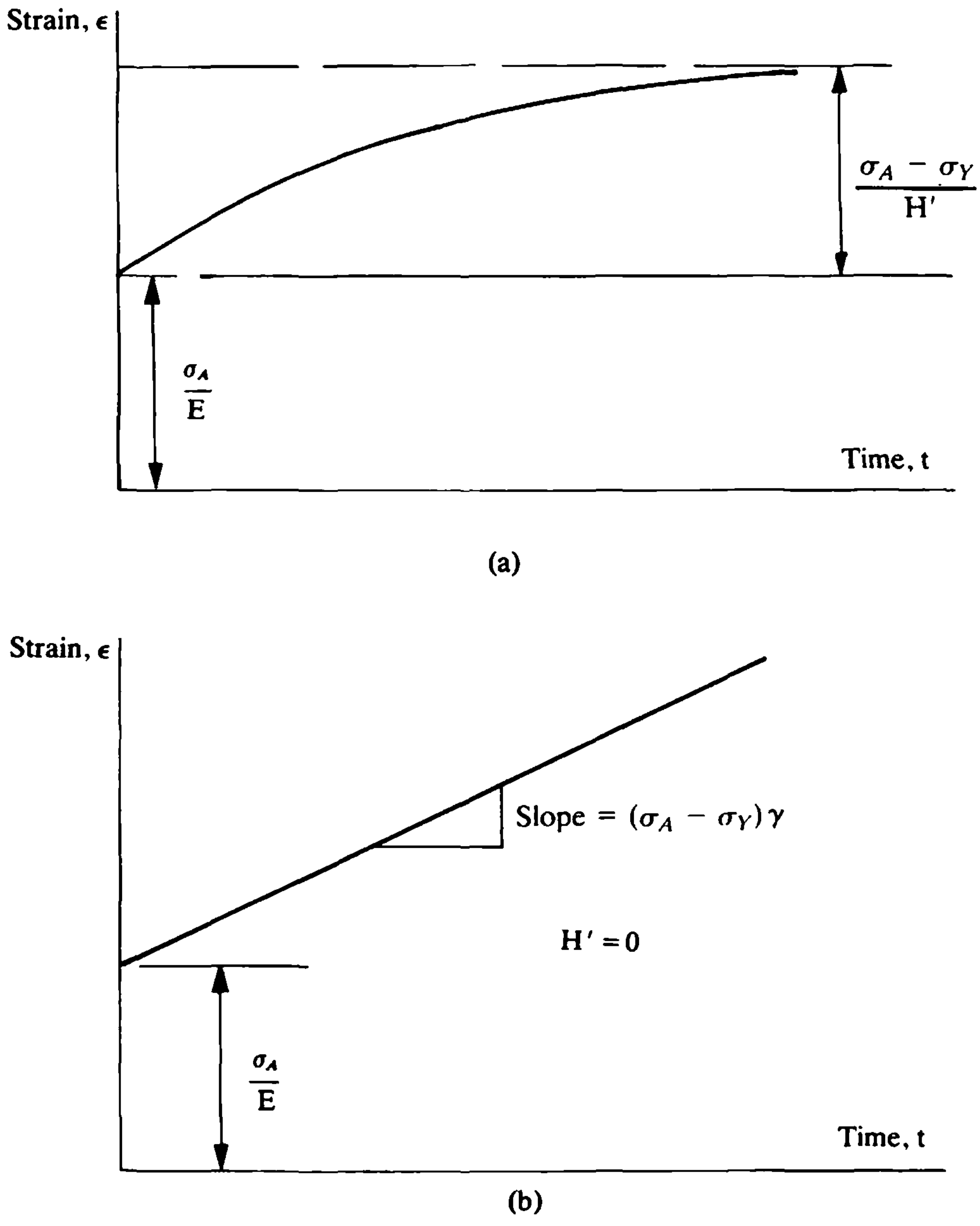


Fig. 4.2 Strain response with time for the model of Fig. 4.1 due to a constant applied load. (a) Linear strain hardening material. (b) Perfectly plastic material.

provided that H' is nonzero. The form of the response is shown in Fig. 4.2(a). Following an initial elastic response, the strain in the model attains the steady state value indicated in an exponential fashion.

The case of a perfectly viscoplastic material in which $H' = 0$, can be obtained by taking the limit as H' tends to zero in (4.16) and applying L'Hopital's rule. This results in

$$\epsilon = \frac{\sigma_A}{E} + (\sigma_A - \sigma_Y)\gamma t. \tag{4.17}$$

This response is shown in Fig. 4.2(b). In this case it is seen that a steady state condition is not achieved and that viscoplastic deformation continues indefinitely at a constant strain rate. The different behaviour shown in Figs. 4.2(a) and 4.2(b) arises from the fact that for a strain hardening material the viscoplastic yield stress increases according to (4.3) until it reaches the applied stress level σ_A at which stage the viscoplastic strain rate becomes zero. On the other hand, for a perfectly viscoplastic material there is always a stress imbalance of $\sigma_A - \sigma_Y$ in the system which does not reduce and consequently steady state conditions cannot be achieved.

We note that in (4.16) and (4.17) that the time t only enters the expressions through the term γt. Therefore the solution for a material with a different fluidity parameter γ can be obtained by a simple adjustment of the time scale.

4.3 Numerical solution process

Viscoplasticity is a transient phenomenon and therefore the essential objective of a numerical solution process is to determine the displacement, strains and stresses throughout the time interval of interest. Consequently some *time stepping* or *time marching* scheme must be introduced in order to allow the solution to be advanced from a time t_n to time $t_{n+1} = t_n + \Delta t_n$, where subscripts n and $n+1$ denote successive times and Δt_n the interval between. The simplest method of incrementing quantities over a time interval is afforded by *Euler's rule.* In this the mean rate of change over the interval is taken as the value at the beginning of the interval and thus the predicted value of some quantity X at time t_{n+1} is extrapolated from the value at time t_n to be

$$X^{n+1} = X^n + (\dot{X})^n \Delta t_n. \tag{4.18}$$

This scheme becomes unstable for time steps exceeding a critical value and estimation of the limiting step length is discussed in Section 4.4. The Euler method, however, remains attractive due to its simplicity.

With the viscoplastic strain rate defined by (4.14) we can define the strain increment $\Delta\epsilon_{vp}{}^n$ occurring in a time interval $\Delta t_n = t_{n+1} - t_n$, using (4.18), as

$$\Delta\epsilon_{vp}{}^n = \dot{\epsilon}_{vp}{}^n \Delta t_n. \tag{4.19}$$

We note that the time step length can, in general, be different for each time interval.

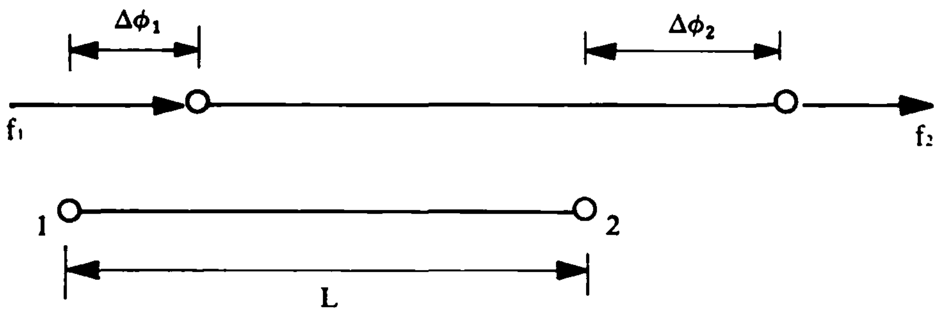


Fig. 4.3 One-dimensional two-noded element with linear displacement variation.

With reference to Fig. 4.3, consider the behaviour of a linear displacement element, which is of length L and has a cross-sectional area, A. The change of length in this element associated with strain increment (4.19) is

$$\Delta\phi^n = \Delta\epsilon_{vp}{}^n L, \tag{4.20}$$

or adding the displacement change due to a change in applied loading Δf^n occurring between times t_n and t_{n+1} we obtain the total change in element length to be

$$\Delta\phi^n = \Delta\epsilon_{vp}{}^n L + \frac{L}{AE}\Delta f^n. \tag{4.21}$$

This can be rewritten in matrix form, in terms of the nodal displacements and forces as

$$\Delta\varphi^n = [K]^{-1}\Delta V^n, \tag{4.22}$$

where

$$\Delta\varphi^n = \begin{bmatrix}\Delta\phi_1{}^n \\ \Delta\phi_2{}^n\end{bmatrix}, \tag{4.23}$$

$$\Delta V^n = AE\dot{\epsilon}_{vp}{}^n\Delta t_n\begin{bmatrix}1 \\ -1\end{bmatrix} + \Delta f^n, \tag{4.24}$$

and

$$K^{(e)} = \frac{EA}{L}\begin{bmatrix}1 & -1 \\ -1 & 1\end{bmatrix}. \tag{4.25}$$

In the above, ΔV^n are termed the *pseudo forces* and $\Delta\varphi^n$ and Δf^n are respectively the incremental changes in the nodal displacements and applied forces for the element.

We note in passing that expressions (4.24) and (4.25) could be written in the standard finite element form

$$\Delta \boldsymbol{V}^n = \int_V \boldsymbol{B}^T \boldsymbol{D} \boldsymbol{\epsilon} \, dV + \Delta \boldsymbol{f}^n$$

$$\boldsymbol{K}^{(e)} = \int_V \boldsymbol{B}^T \boldsymbol{D} \boldsymbol{B} \, dV, \tag{4.26}$$

since for the linear element considered

$$\boldsymbol{B} = \left[-\frac{1}{L}, \frac{1}{L}\right]$$

$$\boldsymbol{D} = E$$

$$\int_V dV = AL. \tag{4.27}$$

The displacements at time t_{n+1} are then obtained by simple accumulation as

$$\varphi^{n+1} = \varphi^n + \Delta\varphi^n. \tag{4.28}$$

The stress increment is given from (4.1) and (4.7) to be

$$\Delta\sigma^n = E\Delta\epsilon_e{}^n = E(\Delta\epsilon^n - \Delta\epsilon_{vp}{}^n), \tag{4.29}$$

or

$$\Delta\sigma^n = E\left(\frac{\Delta\phi_1{}^n - \Delta\phi_2{}^n}{L} - \dot{\epsilon}_{vp}{}^n \Delta t_n\right), \tag{4.30}$$

where $\Delta\phi_1{}^n$ and $\Delta\phi_2{}^n$ are the displacement changes at the nodes of the element.

The stress at time t_{n+1} is then given by

$$\sigma^{n+1} = \sigma^n + \Delta\sigma^n. \tag{4.31}$$

The total viscoplastic strain at time t_{n+1} is

$$\epsilon_{vp}{}^{n+1} = \epsilon_{vp}{}^n + \Delta\epsilon_{vp}{}^n, \tag{4.32}$$

and finally the viscoplastic strain rate at t_{n+1} is given, from (4.14) as

$$\dot{\epsilon}_{vp}{}^{n+1} = \gamma[\sigma^{n+1} - (\sigma_Y + H'\epsilon_{vp}{}^{n+1})]. \tag{4.33}$$

In employing the Euler scheme for time-stepping, we are effectively linearising the variation of quantities over the increment. Therefore the total stresses σ^{n+1} obtained by accumulating all such stress increments may not be in exact equilibrium with the applied forces. It is therefore necessary to introduce an *equilibrium correction* procedure into the numerical solution algorithm. The simplest approach is to evaluate the out-of-balance nodal forces at the end of each time step and consider these forces as additional forces to be applied at the beginning of the next time increment.

The out-of-balance or residual forces, ψ, for the general element are given as the algebraic sum of the applied nodal loads and the nodal forces equivalent to the element stress, so that

$$\psi^{n+1} = A\sigma^{n+1}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + f^{n+1}, \tag{4.34}$$

in which σ^{n+1} is the element stress and f^{n+1} are the total applied forces at time t_{n+1}. These residual forces are then added to the pseudo forces to give for the next time increment

$$\Delta V^{n+1} = AE\,\dot{\epsilon}_{vp}{}^{n+1}\,\Delta t_{n+1}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \Delta f^{n+1} + \psi^{n+1}. \tag{4.35}$$

This sequence is repeated for each time step until solution is either obtained for the desired time duration or until steady state conditions are achieved. Steady state conditions are deemed to have been achieved when the viscoplastic strain rate, $\dot{\epsilon}_{vp}{}^{n}$, becomes tolerably small.

4.4 Limiting time-step length

The critical time-step length for viscoplastic solution using the Euler time marching scheme has been established by Cormeau.[1] For the uniaxial case considered in this chapter the limiting value is

$$\Delta t \leqslant \frac{\sigma_Y}{\gamma E}. \tag{4.36}$$

Alternatively the time-step length can be limited according to a semi-empirical relationship. Such an approach is essential for some general continuum problems where a theoretical value of the critical time-step length may not exist. The most obvious procedure is to limit the viscoplastic strain increment to be some specified factor, τ, of the total current strain,

$$\dot{\epsilon}_{vp}{}^{n}\,\Delta t_n \leqslant \tau\epsilon^n. \tag{4.37}$$

Since each element generally has a different strain level, expression (4.37) will yield a different limiting step value when applied to each element in turn. Therefore the limiting value is restricted according to

$$\Delta t_n \leqslant \tau\left[\frac{\epsilon^n}{\dot{\epsilon}_{vp}{}^{n}}\right]_{\min}, \tag{4.38}$$

where the minimum value of Δt_n obtained after considering each element is taken. Stability of the solution process is also aided by restricting the length of successive time steps according to

$$\Delta t_{n+1} \leqslant k\Delta t_n, \tag{4.39}$$

where k is a specified constant generally chosen in the range 1·5–2·0.

4.5 Computational procedure

Before proceeding with the development of a computer code for the solution of one-dimensional viscoplastic problems we will first summarise the essential steps of the computation. Solution to the problem must commence from the known initial conditions at time $t = 0$ which of course correspond to the initial elastic response. At this stage φ^0, f^0, ϵ^0, σ^0 are known and $\epsilon_{vp}{}^0 = 0$. The general procedure for advancing the solution from a time t_n to time t_{n+1} is the following.

Stage 1 At time $t = t_n$ the values of σ^n, ϵ^n, $\epsilon_{vp}{}^n$ and f^n are known for each element and the nodal displacements are also known. The viscoplastic strain rate for each element is then evaluated according to (4.14) as

$$\dot{\epsilon}_{vp}{}^n = \gamma[\sigma^n - (\sigma_Y + H'\,\epsilon_{vp}{}^n)]. \tag{4.40}$$

Stage 2 (a) Compute the displacement increments, $\Delta\varphi^n$, according to (4.22)–(4.25), as

$$\Delta\varphi^n = [K]^{-1}\Delta V^n,$$

where

$$\Delta V^n = AE\,\dot{\epsilon}_{vp}{}^n\,\Delta t_n\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \Delta f^n,$$

and the stiffness matrix for an individual element is

$$K^{(e)} = \frac{EA}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

(b) Calculate the stress increment $\Delta\sigma^n$ and the viscoplastic strain increment $\Delta\epsilon_{vp}{}^n$ for each element as

$$\Delta\sigma^n = E\left(\frac{\Delta\phi_1{}^n - \Delta\phi_2{}^n}{L} - \dot{\epsilon}_{vp}{}^n\,\Delta t_n\right),$$

$$\Delta\epsilon_{vp}{}^n = \dot{\epsilon}_{vp}{}^n\,\Delta t_n.$$

Stage 3 Determine the total displacements, stresses and viscoplastic strain

$$\varphi^{n+1} = \varphi^n + \Delta\varphi^n,$$

$$\sigma^{n+1} = \sigma^n + \Delta\sigma^n,$$

$$\epsilon_{vp}{}^{n+1} = \epsilon_{vp}{}^n + \Delta\epsilon_{vp}{}^n.$$

Stage 4 Calculate the viscoplastic strain rate for each element

$$\dot{\epsilon}_{vp}{}^{n+1} = \gamma[\sigma^{n+1} - (\sigma_Y + H'\,\epsilon_{vp}{}^{n+1})].$$

Stage 5 Apply the equilibrium correction. Evaluate the residual forces, for each element, as

$$\psi^{n+1} = A\sigma^{n+1}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + f^{n+1}.$$

Add these into the vector of incremental pseudo loads for use in the next time step

$$\Delta V^{n+1} = AE\dot{\epsilon}_{vp}{}^{n+1}\,\Delta t_{n+1}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \Delta f^{n+1} + \psi^{n+1}.$$

Stage 6 Check to see if the viscoplastic strain rate $\dot{\epsilon}_{vp}{}^{n+1}$ in each element has become tolerably small. If so, steady state conditions have been reached and the solution is either terminated or the next load increment is applied. If $\dot{\epsilon}_{vp}{}^{n+1}$ is non-zero return to Stage 1 and repeat the entire procedure for the next time step.

4.6 Program structure

The organisation of the one-dimensional viscoplastic program is shown in Fig. 4.4 where, in particular, the order in which subroutines are accessed is indicated. The operations undertaken by the program are those described in Section 4.5. Many of the subroutines employed are common to the one-dimensional plasticity application described in Chapter 3 and, since they are used in the present program without modification, the reader will be referred to the appropriate section for details. Only the additional subroutines necessary to complete the computer package will be described in this chapter.

With reference to Fig. 4.4 the following subroutines have been already described where indicated below:

Subroutine ASSEMB —Section 3.4.2
Subroutine GREDUC—Section 3.4.3
Subroutine BAKSUB —Section 3.4.4
Subroutine RESOLV —Section 3.4.5
Subroutine RESULT —Section 3.5
Subroutine INITAL —Section 3.6*

Also, Subroutine DATA described in Section 3.2 is used with some minor modifications. A viscoplastic material in one dimension requires five individual quantities to describe it completely. Thus NPROP becomes 5 and the following quantities must be specified as material properties.

PROPS (NUMAT, 1)—The elastic modulus, E, of the material
PROPS (NUMAT, 2)—The cross-sectional area, A, of the element
PROPS (NUMAT, 3)—The uniaxial yield stress, σ_Y, of the material
PROPS (NUMAT, 4)—The linear strain hardening parameter, H', for the material
PROPS (NUMAT, 5)—The fluidity parameter, γ, controlling the viscoplastic strain rate.

* Subroutine NONAL, described in Section 3.3, is also employed but with IITER now replaced by the time step index, ISTEP.

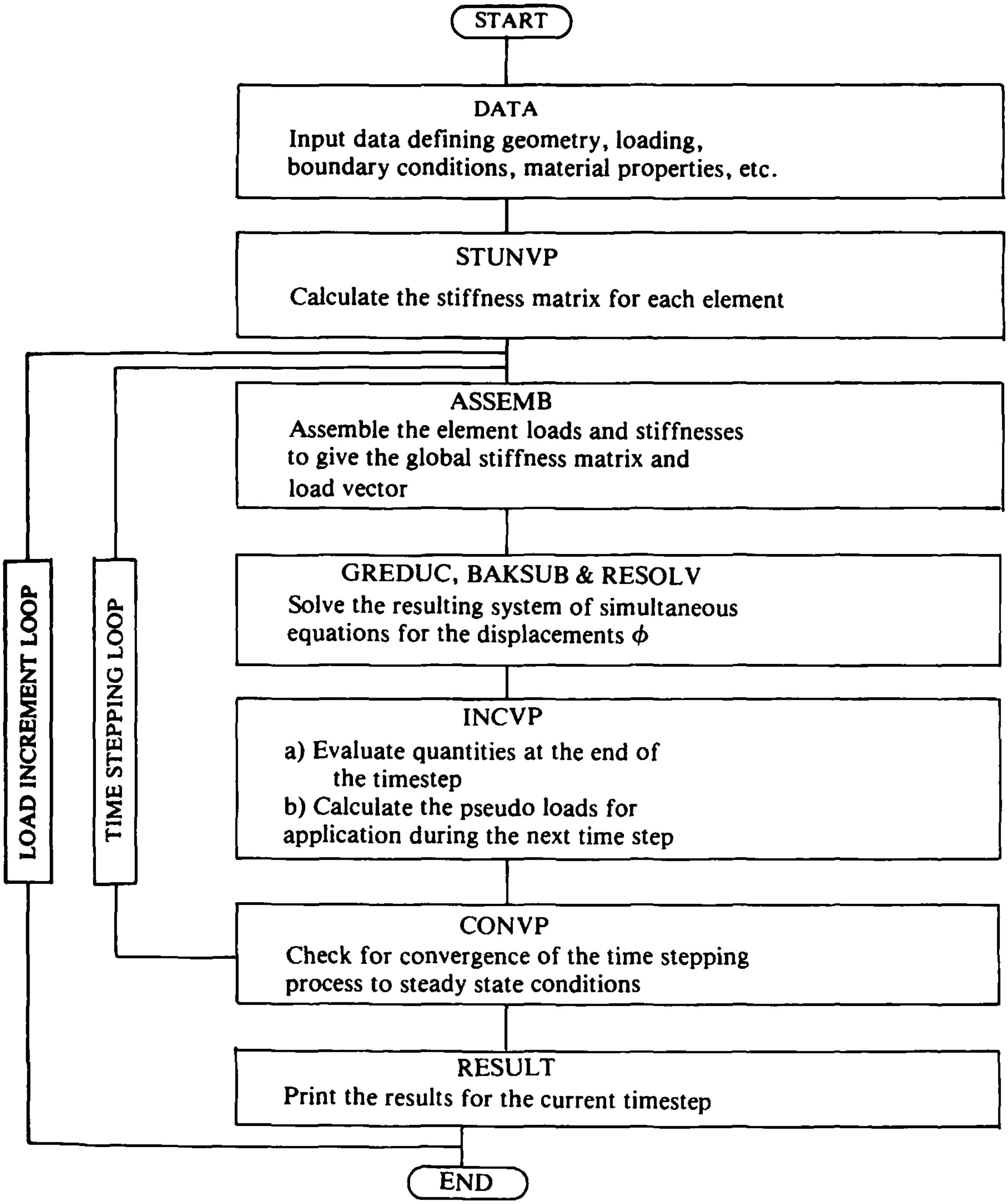


Fig. 4.4 Operational sequence for the one-dimensional viscoplastic stress analysis program.

Input data are also received by this segment which controls the time-stepping algorithm. The following information is input:

TAUFT	The parameter τ discussed in Section 4.4
DTINT	The time-step length for the first time step
FTIME	The factor k defined in (4.39) which limits the relative length of successive time steps

The additional subroutines which are required will now be described in turn.

4.7 Element stiffness subroutine STUNVP

In all stages of the viscoplastic solution the elastic element stiffness matrix is employed, as indicated in (4.25). Consequently the structure of subroutine STUNVP, which evaluates the stiffness matrix for each element in turn, is straightforward and can be presented without further comment.

```
      SUBROUTINE  STUNVP                                                SNVP   1
C***********************************************************************SNVP   2
C                                                                       SNVP   3
C *** CALCULATES ELEMENT STIFFNESS MATRICES                             SNVP   4
C                                                                       SNVP   5
C***********************************************************************SNVP   6
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,ISTEP,     SNVP   7
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     SNVP   8
     .             NSTEP,NOUTP,FACTO,TAUFT,DTINT,FTIME,FIRST,PVALU,     SNVP   9
     .             DTIME,TTIME                                          SNVP  10
      COMMON/UNIM2/PROPS(5,5),COORD(26),LNODS(25,2),IFPRE(52),          SNVP  11
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       SNVP  12
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           SNVP  13
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      SNVP  14
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),VIVEL(25) SNVP  15
      REWIND 1                                                          SNVP  16
      DO 10 IELEM=1,NELEM                                               SNVP  17
      LPROP=MATNO(IELEM)                                                SNVP  18
      YOUNG=PROPS(LPROP,1)                                              SNVP  19
      XAREA=PROPS(LPROP,2)                                              SNVP  20
      NODE1=LNODS(IELEM,1)                                              SNVP  21
      NODE2=LNODS(IELEM,2)                                              SNVP  22
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))                              SNVP  23
      FMULT=YOUNG*XAREA/ELENG                                           SNVP  24
      ESTIF(1,1)=FMULT                                                  SNVP  25
      ESTIF(1,2)=-FMULT                                                 SNVP  26
      ESTIF(2,1)=-FMULT                                                 SNVP  27
      ESTIF(2,2)=FMULT                                                  SNVP  28
      WRITE(1) ESTIF                                                    SNVP  29
   10 CONTINUE                                                          SNVP  30
      RETURN                                                            SNVP  31
      END                                                               SNVP  32
```

SNVP 16 Rewind the file on which the stiffness matrix of each element will be stored.

SNVP 17 Loop over each element.

SNVP 18 Identify the material property of the current element.

SNVP 19–20 Set YOUNG equal to the material elastic modulus and XAREA equal to the cross-sectional area.

SNVP 21–22 Identify the node numbers of the element.

SNVP 23 Calculate the element length.

SNVP 24 Compute EA/L as FMULT.

SNVP 25–28 Evaluate the components of the element stiffness matrix according to (4.25).

SNVP 29 Write the element stiffness matrix on to disc file.

SNVP 30 End of loop over each element.

4.8 Subroutine INCVP for the evaluation of end of time-step quantities and equilibrium correction terms

This subroutine evaluates quantities such as stresses and viscoplastic strains at the end of the current time step and also calculates the loading to be applied during the next time step. Essentially it undertakes Stages 3–5 described in Section 4.5. All quantities at the end of time step n are calculated as $(\ )^{n+1}$.

The program presented is restricted to loading which is applied in discrete increments and is assumed to remain constant during the time-stepping process for any given increment. Thus in (4.35) $\Delta f^n = 0$ for all stages other than the first time step of a particular load increment.

Subroutine INCVP is now presented and described.

```
      SUBROUTINE INCVP                                                  INVP
C***********************************************************************INVP
C                                                                       INVP
C *** CALCULATES INTERNAL EQUIVALENT NODAL FORCES                       INVP
C                                                                       INVP
C***********************************************************************INVP
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,ISTEP,     INVP
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     INVP
     .             NSTEP,NOUTP,FACTO,TAUFT,DTINT,FTIME,FIRST,PVALU,     INVP
     .             DTIME,TTIME                                          INVP
      COMMON/UNIM2/PROPS(5,5),COORD(26),LNODS(25,2),IFPRE(52),          INVP
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       INVP
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           INVP
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      INVP
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),VIVEL(25) INVP
      DO 10 IELEM=1,NELEM                                               INVP
      DO 10 IEVAB=1,NEVAB                                               INVP
   10 ELOAD(IELEM,IEVAB)=0.0                                            INVP
      DNEXT=FTIME*DTIME                                                 INVP
      DO 30 IELEM=1,NELEM                                               INVP
      LPROP=MATNO(IELEM)                                                INVP
      YOUNG=PROPS(LPROP,1)                                              INVP
      XAREA=PROPS(LPROP,2)                                              INVP
      YIELD=PROPS(LPROP,3)                                              INVP
      HARDS=PROPS(LPROP,4)                                              INVP
      GAMMA=PROPS(LPROP,5)                                              INVP
      NODE1=LNODS(IELEM,1)                                              INVP
      NODE2=LNODS(IELEM,2)                                              INVP
      ELENG=ABS(COORD(NODE1)-COORD(NODE2))                              INVP
      IF(COORD(NODE2).GT.COORD(NODE1)) STRAN=(XDISP(NODE2)-XDISP(NODE1))INVP
     . /ELENG                                                           INVP
      IF(COORD(NODE2).LT.COORD(NODE1)) STRAN=(XDISP(NODE1)-XDISP(NODE2))INVP
     . /ELENG                                                           INVP
      STRES(IELEM,1)=STRES(IELEM,1)+(STRAN-VIVEL(IELEM)*DTIME)*YOUNG    INVP
      PLAST(IELEM)=PLAST(IELEM)+VIVEL(IELEM)*DTIME                      INVP
      IF(STRES(IELEM,1).LT.0.0) YIELD=-YIELD                            INVP
      PREYS=YIELD+HARDS*PLAST(IELEM)                                    INVP
      IF(ABS(STRES(IELEM,1)).LE.ABS(PREYS)) GO TO 20                    INVP
      VIVEL(IELEM)=GAMMA*(STRES(IELEM,1)-(YIELD+HARDS*PLAST(IELEM)))    INVP
      SNTOT=(TDISP(NODE2,1)-TDISP(NODE1,1))/ELENG                       INVP
      DELTM=TAUFT*ABS(SNTOT/VIVEL(IELEM))                               INVP
      IF(DELTM.LT.DNEXT) DNEXT=DELTM                                    INVP
      GO TO 30                                                          INVP
   20 VIVEL(IELEM)=0.0                                                  INVP
```

```
   30 CONTINUE                                                          INVP  45
      DTIME=DNEXT                                                       INVP  46
      IF(ISTEP.EQ.1) DTIME=DTINT                                        INVP  47
      DO 50 IELEM=1,NELEM                                               INVP  48
      LPROP=MATNO(IELEM)                                                INVP  49
      YOUNG=PROPS(LPROP,1)                                              INVP  50
      XAREA=PROPS(LPROP,2)                                              INVP  51
      FACTR=(YOUNG*VIVEL(IELEM)*DTIME-STRES(IELEM,1))*XAREA             INVP  52
      IF(COORD(NODE2).GT.COORD(NODE1)) GO TO 40                         INVP  53
      ELOAD(IELEM,1)= FACTR                                             INVP  54
      ELOAD(IELEM,2)=-FACTR                                             INVP  55
      GO TO 50                                                          INVP  56
   40 ELOAD(IELEM,1)=-FACTR                                             INVP  57
      ELOAD(IELEM,2)= FACTR                                             INVP  58
   50 CONTINUE                                                          INVP  59
      DO 60 IELEM=1,NELEM                                               INVP  60
      DO 60 IEVAB=1,NEVAB                                               INVP  61
   60 ELOAD(IELEM,IEVAB)=ELOAD(IELEM,IEVAB)+TLOAD(IELEM,IEVAB)          INVP  62
      RETURN                                                            INVP  63
      END                                                               INVP  64
```

INVP 16–18 Zero the array in which the pseudo loads for the next time step will be stored.

INVP 20 Loop over each element.

INVP 21 Identify the element material property number.

INVP 22–26 Store the elastic modulus as YOUNG, the cross-sectional area as XAREA, the uniaxial yield stress as YIELD, the uniaxial hardening parameter as HARDS and the fluidity parameter as GAMMA.

INVP 27–28 Identify the element node numbers.

INVP 29 Evaluate the length of the element.

INVP 30–33 Calculate the element strain so that a tensile strain is positive.

INVP 34 Evaluate the total current stress σ^{n+1} according to (4.30) and (4.31).

INVP 35 Evaluate the total viscoplastic strain $\epsilon_{vp}{}^{n+1}$, according to (4.32).

INVP 36 For a compressive stress take a negative value of the initial yield stress.

INVP 37 Compute the current yield level $\sigma_Y + H' \epsilon_{vp}{}^{n+1}$.

INVP 38 If the current stress is less than the current yield stress, avoid evaluation of the viscoplastic strain rate.

INVP 39 Otherwise evaluate the viscoplastic strain rate according to (4.33).

INVP 40–42 Evaluate the next time-step length according to (4.38).

INVP 44 For elastic elements set the viscoplastic strain rate to zero.

INVP 45 End of element loop.

INVP 47 For the first timestep of a load increment choose the timestep as the initial value.

INVP 48 Enter element loop to evaluate pseudo loads, $\Delta V^{n+1}_{\cdot}$, for the next time step.

INVP 49 Identify the element material property number.

INVP 50–51 Store the elastic modulus as YOUNG and the cross-sectional area as XAREA.

INVP 52 Evaluate the factor $AE\dot{\epsilon}_{vp}{}^{n+1}\Delta t_{n+1}+A\sigma^{n+1}$.

INVP 53–62 Evaluate ΔV^{n+1} according to (4.34) and (4.35), taking the appropriate signs for tensile or compressive stresses and strains. Note that $f^{n+1}+\Delta f^{n+1}$ is the total load applied for time step $n+1$ which is stored as TLOAD.

4.9 Convergence monitoring subroutine, CONVP

Convergence of the numerical process to the steady state solution must be monitored by comparing, in some way, the values of the viscoplastic strain rate determined during each time step. This can be done in several ways and in this section we describe a procedure based on a *global* convergence check only. In particular we will assume that steady state conditions have been achieved if

$$\frac{\sum_{i=1}^{M}|(\Delta\epsilon_{vp}{}^{n})_i|}{\sum_{i=1}^{M}|(\Delta\epsilon_{vp}{}^{1}{}_{i})|}\times 100 \leqslant \text{TOLER}, \tag{4.41}$$

where M denotes the total number of elements in the problem and $|\ |$ denotes the absolute value. The multiplication factor of 100 on the left-hand side allows the specified tolerance factor TOLER to be considered as a percentage term. Equation (4.41) states that steady state conditions are deemed to have been achieved if the sum of the absolute values of the strain increment for any time step is less than or equal to TOLER times the corresponding value for the first time step. For practical purposes a value of TOLER $\leqslant$ 1·0 (i.e. 1%) is generally adequate. Parameter NCHEK indicates convergence of the solution to steady state, where;

NCHEK = 1 indicates that the solution is converging to steady state, with the viscoplastic strain increment reducing between two successive time steps.

NCHEK = 999 indicates a divergence, with the viscoplastic strain increment increasing between two successive time steps.

NCHEK = 0 indicates that steady state conditions have been achieved.

Subroutine CONVP is now presented and described.

```
      SUBROUTINE  CONVP
C*************************************************************************
C
C *** CHECKS FOR SOLUTION CONVERGENCE
C
C*************************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,ISTEP,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
```

```
     .         NSTEP,NOUTP,FACTO,TAUFT,DTINT,FTIME,FIRST,PVALU,       CNVP   9
     .         DTIME,TTIME                                            CNVP  10
      COMMON/UNIM2/PROPS(5,5),COORD(26),LNODS(25,2),IFPRE(52),        CNVP  11
     .         FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),         CNVP  12
     .         MATNO(25),STRES(25,2),PLAST(25),XDISP(52),             CNVP  13
     .         TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),        CNVP  14
     .         REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),VIVEL(25)   CNVP  15
      NCHEK=1                                                         CNVP  16
      TOTAL=0.0                                                       CNVP  17
      DO 10 IELEM=1,NELEM                                             CNVP  18
   10 TOTAL=TOTAL+ABS(VIVEL(IELEM))*DTIME                             CNVP  19
      IF(ISTEP.EQ.1) FIRST=TOTAL                                      CNVP  20
      IF(FIRST.EQ.0.0) GO TO 20                                       CNVP  21
      RATIO=100.0*TOTAL/FIRST                                         CNVP  22
      GO TO 30                                                        CNVP  23
   20 RATIO=0.0                                                       CNVP  24
   30 CONTINUE                                                        CNVP  25
      IF(RATIO.LE.TOLER) NCHEK=0                                      CNVP  26
      IF(RATIO.GT.PVALU) NCHEK=999                                    CNVP  27
   40 PVALU=RATIO                                                     CNVP  28
      WRITE(6,900) TTIME                                              CNVP  29
  900 FORMAT(1H0,5X,12HTOTAL TIME =,E17.6)                            CNVP  30
      WRITE(6,910) NCHEK,RATIO                                        CNVP  31
  910 FORMAT(1H0,5X,18HCONVERGENCE CODE =,I4,3X,28HNORM OF RESIDUAL SUM CNVP  32
     .RATIO =,E14.6)                                                  CNVP  33
      RETURN                                                          CNVP  34
      END                                                             CNVP  35
```

CNVP 16 Set the indicator monitoring convergence to 1. This will be reset later in the subroutine if necessary.

CNVP 17–19 Compute

$$\sum_{i=1}^{M} |(\Delta\epsilon_{vp}{}^{n})_i|$$

for the current time step as required in (4.41).

CNVP 20 For the first time step evaluate the denominator in (4.41).

CNVP 21–25 Evaluate the left-hand side in (4.41). If the denominator is zero there is no viscoplastic flow for the particular load increment, therefore set RATIO = 0 indicating a steady state condition.

CNVP 26 If (4.41) is satisfied, set NCHEK = 0 indicating a steady state condition.

CNVP 27 If the viscoplastic increment has increased from the value obtained on the previous time step set NCHEK = 999.

CNVP 28 Store the current value of the left-hand side of (4.41) for use in Statement CNVP 27 during the next time step.

CNVP 29–30 Output the current time.

CNVP 31–33 Output the value of NCHEK and the left-hand side of (4.41).

4.10 Subroutine INCLOD

Subroutine INCLOD described in Section 3.7 is employed for this application with one minor change: The iteration limit NITER is now replaced by the time-step limit NSTEP.

For each increment of load, data is accepted by INCLOD to control the upper limit to the number of time steps, the output frequency, the size of load increment and the convergence tolerance limit. These quantities are specifically input as:

NSTEP Maximum permissible number of time steps. This is a safety measure to cover situations where steady state conditions are not achieved. After performing NSTEP time steps the program will then stop.

NOUTP This parameter controls the frequency of output of results:
0—Print the results on convergence to steady state conditions only, for each load increment.
1—Print the results after the first time step *and* at steady state, for each load increment.
2—Print the results for each time step for each load increment.

FACTO This quantity controls the magnitude of any load increment. The applied loading is accepted by subroutine DATA and stored in array RLOAD. The size of any load increment is then RLOAD factored by FACTO. Therefore if FACTO is input for the first three increments as respectively 0·3, 0·3 and 0·1, the total loading applied to the structure during the third increment is 0·7 times the loading input in subroutine DATA.

TOLER This item of data controls the tolerance permitted on the steady state convergence process, and has been described in Section 4.9.

Subject to the replacement of NITER by NSTEP, the form of this subroutine for the present application is identical to that provided in Section 3.7.

4.11 The main, master or controlling segment

This master segment controls the calling, in order, of the other subroutines. This program segment also controls the time-stepping process and also the incrementing of the applied loads, where appropriate.

The following channel numbers are employed by the program: 5 (card reader), 6 (line printer), 1 (scratch file).

```
      MASTER UNVISC
C***********************************************************************
C
C *** PROGRAM FOR THE 1-D SOLUTION OF NONLINEAR PROBLEMS
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,ISTEP,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NSTEP,NOUTP,FACTO,TAUFT,DTINT,FTIME,FIRST,PVALU,
     .             DTIME,TTIME
      COMMON/UNIM2/PROPS(5,5),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),VIVEL(25)
```

```
      TTIME=0.0                                                    UVIS  16
      CALL  DATA                                                   UVIS  17
      CALL  INITAL                                                 UVIS  18
      CALL  STUNVP                                                 UVIS  19
      DO 30 IINCS=1,NINCS                                          UVIS  20
      CALL  INCLOD                                                 UVIS  21
      DTIME=0.0                                                    UVIS  22
      DO 10 ISTEP=1,NSTEP                                          UVIS  23
      TTIME=TTIME+DTIME                                            UVIS  24
      CALL NONAL                                                   UVIS  25
      CALL  ASSEMB                                                 UVIS  26
      IF(KRESL.EQ.1) CALL GREDUC                                   UVIS  27
      IF(KRESL.EQ.2) CALL RESOLV                                   UVIS  28
      CALL  BAKSUB                                                 UVIS  29
      CALL  INCVP                                                  UVIS  30
      CALL  CONVP                                                  UVIS  31
      IF(NCHEK.EQ.0) GO TO 20                                      UVIS  32
      IF(ISTEP.EQ.1.AND.NOUTP.EQ.1) CALL  RESULT                   UVIS  33
      IF(NOUTP.EQ.2) CALL  RESULT                                  UVIS  34
   10 CONTINUE                                                     UVIS  35
      WRITE(6,900)                                                 UVIS  36
  900 FORMAT(1H0,5X,'STEADY STATE NOT ACHIEVED')                   UVIS  37
      STOP                                                         UVIS  38
   20 CALL  RESULT                                                 UVIS  39
   30 CONTINUE                                                     UVIS  40
      STOP                                                         UVIS  41
      END                                                          UVIS  42
```

UVIS 16 Initialise the total time to zero.

UVIS 17 Call the subroutine which reads the input data as described in Section 3.2.

UVIS 18 Call Subroutine INITAL which:
- (i) Initialises to zero the viscoplastic strain vector and the stress vector.
- (ii) Initialises the array, ELOAD, which will contain the pseudo loads to be applied during each time step.
- (iii) Initialises the vector of applied loads.
- (iv) Initialises the vector of total displacements and total reactions.

UVIS 19 Call the subroutine which evaluates the stiffness matrix for each element.

UVIS 20 Enter the DO LOOP over the number of load increments.

UVIS 21 Call Subroutine INCLOD which:
- (i) Reads and writes the input data required for each load increment as described previously in Section 4.10.
- (ii) Adds the current increment of load into the pseudo load vector, ELOAD, and into the total applied load vector, TLOAD.

UVIS 23 Begin the time-stepping process.

UVIS 24 Calculate the total time elapsed (note that the first time step corresponds to the elastic solution).

UVIS 25 Call the subroutine which sets the parameter KRESL controlling equation resolution facility.

UVIS 26–29 Call the subroutines which assemble the element stiffnesses and solve for the unknown displacements and reactions.

UVIS 30 Call the subroutine which evaluates quantities at the end of the time step and evaluates the loads for the next time step.

UVIS 31 Check whether or not steady state conditions have been achieved.

UVIS 32 If so, terminate the time-stepping process for the current load increment.

UVIS 33–34 Output the results at a frequency controlled by parameter, NOUTP.

UVIS 35 End of time-stepping loop.

UVIS 36–38 If steady state conditions have not been achieved when the upper time-step limit has been reached, write a message and terminate the execution.

UVIS 40 End of load increment loop.

4.12 Numerical examples

The first example considered is the viscoplastic deformation of a single element under constant applied loading. The element is of length 10 units and the applied load is 15 units. The material properties assumed are included in Fig. 4.5, where it is noted that the strain hardening parameter is taken to be zero. The finite element prediction is seen to be in excellent agreement with the theoretical result (4.17) for this problem.

The problem was then reanalysed for a strain-hardening material with $H' = 5000$. The finite element results are compared with the theoretical expression (4.16) in Fig. 4.6 for three different values of the time-stepping parameter, τ, defined in Section 4.4. For a value of $\tau = 0.01$ excellent agreement is obtained, but as the time-step length is increased ($\tau = 0{\cdot}05$ and $\tau = 0{\cdot}1$) comparison with the theoretical solution deteriorates. In particular, an increase in the time-step length progressively overestimates the viscoplastic strain increment, which is a characteristic of the Euler method of time stepping. It is noted that the time-step length is not so critical in the perfectly viscoplastic case of Fig. 4.5 since the exact viscoplastic strain increment is in fact linear for this case.

For the material properties assumed, the theoretical value of the limiting time step is given from (4.36) to be 1·0. It is seen from Figs. 4.5 and 4.6 that the time-step lengths employed in solution are well within this critical value. However, Fig. 4.6 shows that to achieve an accurate result even smaller time-step lengths must be taken. Thus although the theoretical value of the limiting time-step length guarantees *numerical stability* of the solution process it may not always lead to an *accurate* solution.

The second example considered illustrates the redistribution of stress with time which generally takes place in viscoplastic problems. Figure 4.7 shows two members in parallel which are subjected to an end load P which

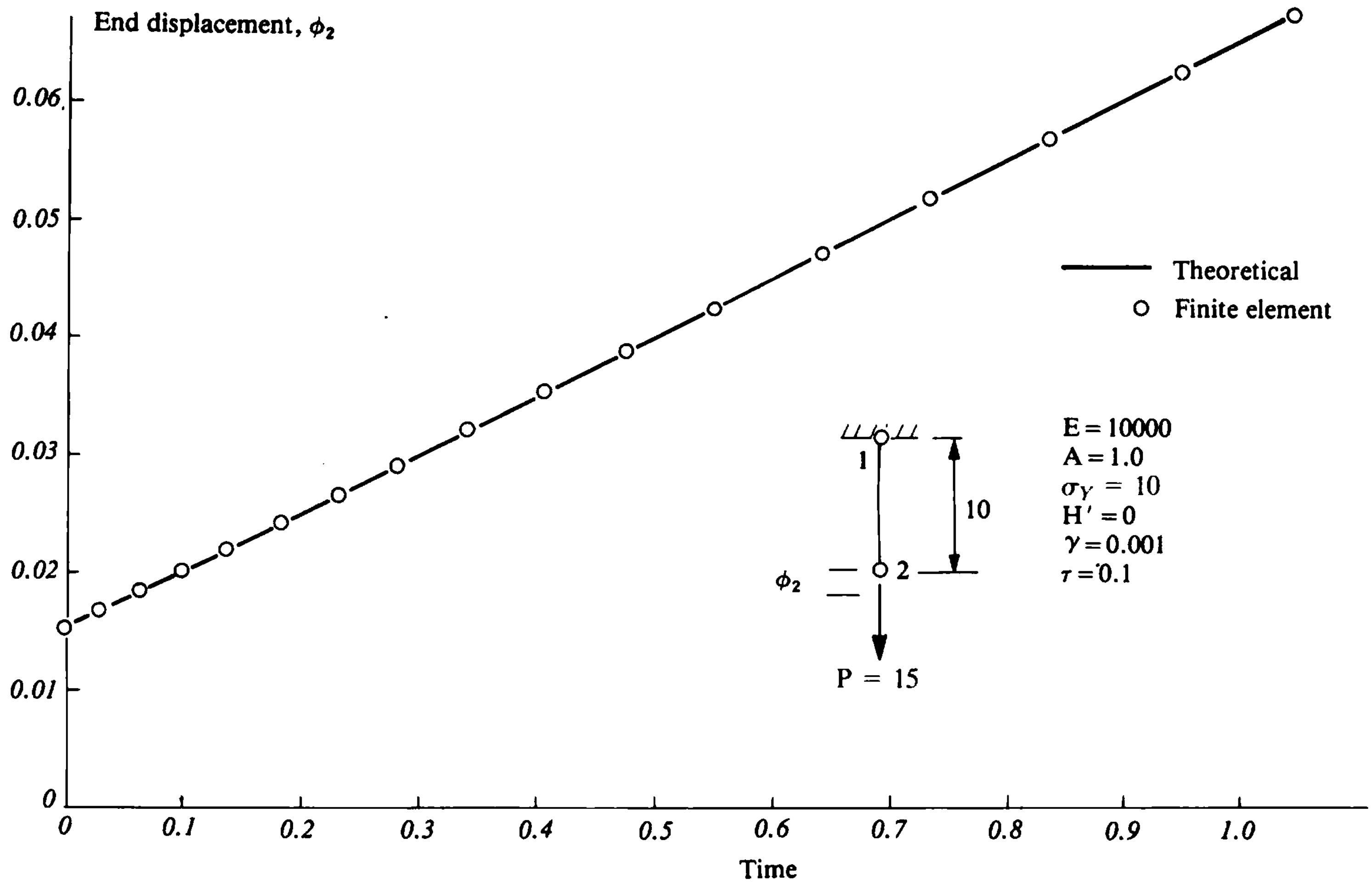


Fig. 4.5 End displacement with time for a single viscoplastic element under constant applied load—No strain hardening.

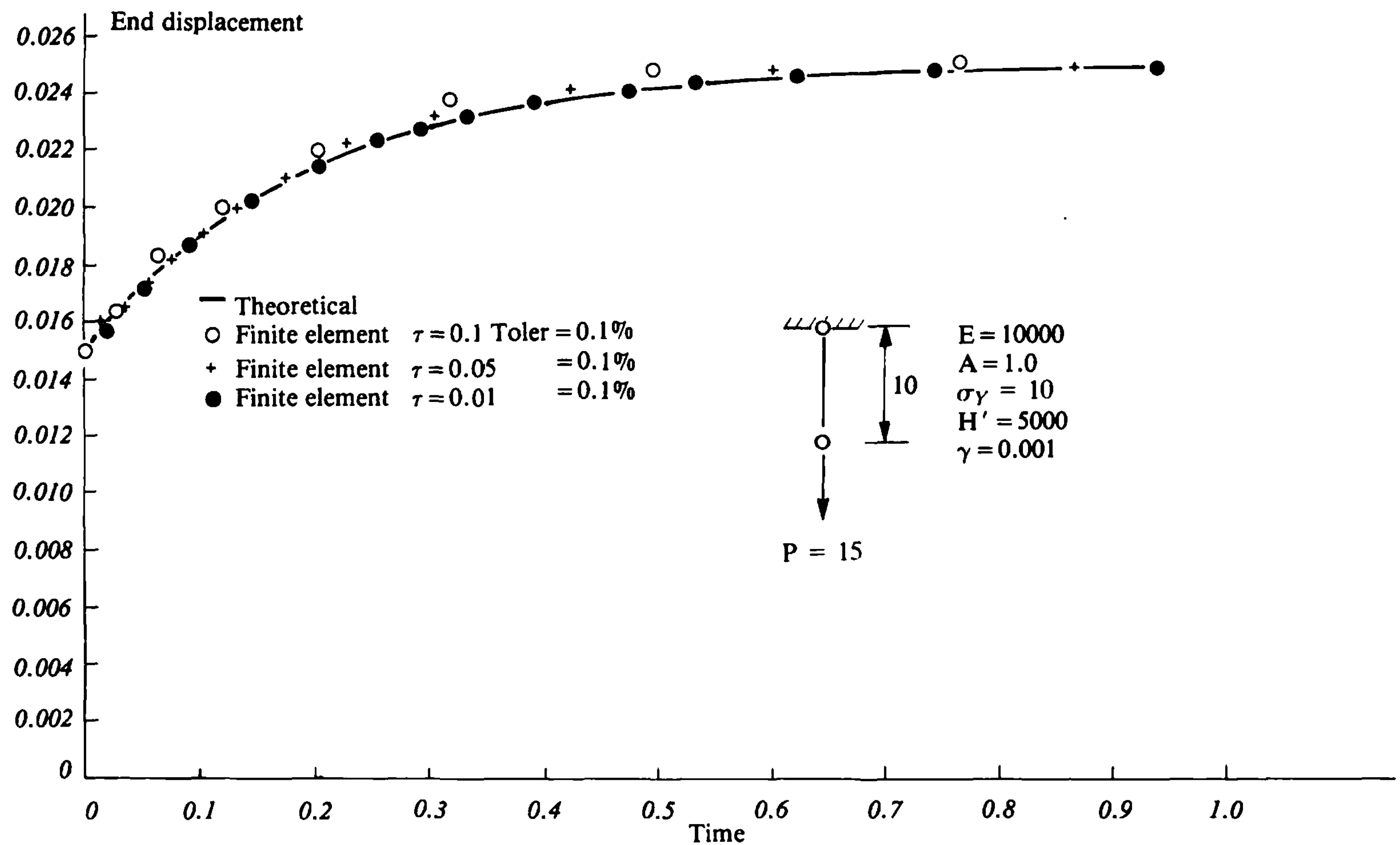


Fig. 4.6 End displacement with time for a single viscoplastic element under constant applied load showing finite element results for different time-step lengths—Linear strain hardening.

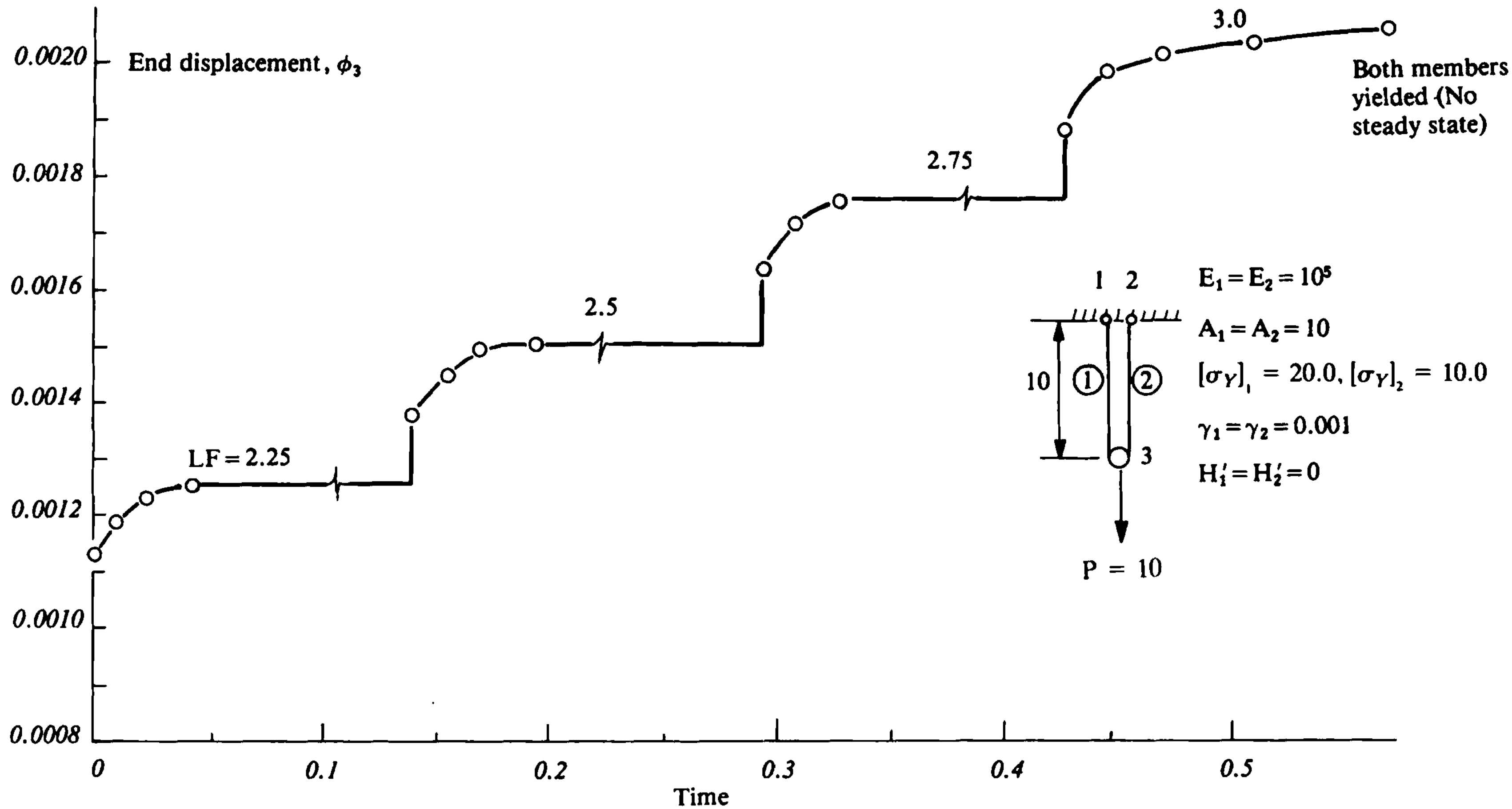


Fig. 4.7 End displacement with time for an elasto-viscoplastic parallel bar model subjected to an incrementally applied end load showing the attainment of steady state conditions.

is incrementally applied. The material properties for each element are included in Fig. 4.7 with the only difference between the two members being the initial yield stress of the materials. The load is applied in four increments and steady state conditions are allowed to develop for each increment before application of further load. The end displacement with time is shown in Fig. 4.7. Steady state conditions are achieved for the first three load increments but not for the fourth since both elements, which behave perfectly plastically, have become yielded at this stage.

4.13 Problems

4.1 Develop the relationship between the applied stress, σ, and the total strain, ϵ, for the rheological model shown in Fig. 4.8. Plot the strain response with time when the model is subjected to a constant applied stress, σ_A.

4.2 Repeat Problem 4.1 for the rheological model shown in Fig. 4.9. In this case the friction slider becomes active for $\sigma \geqslant Y$ where, for a linear strain hardening material, $Y = \sigma_Y + H'\epsilon_{vp}$.

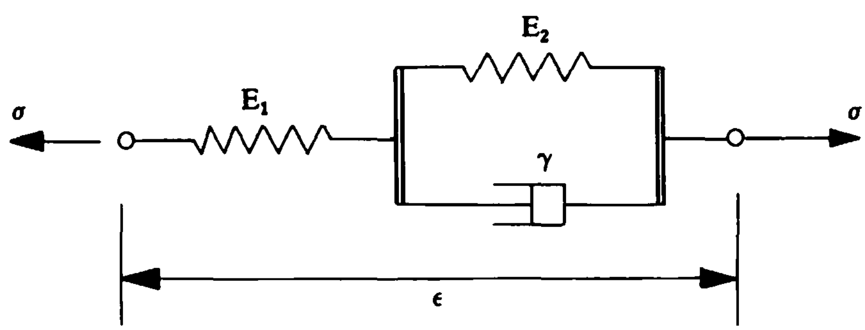


Fig. 4.8 Problem 4.1.

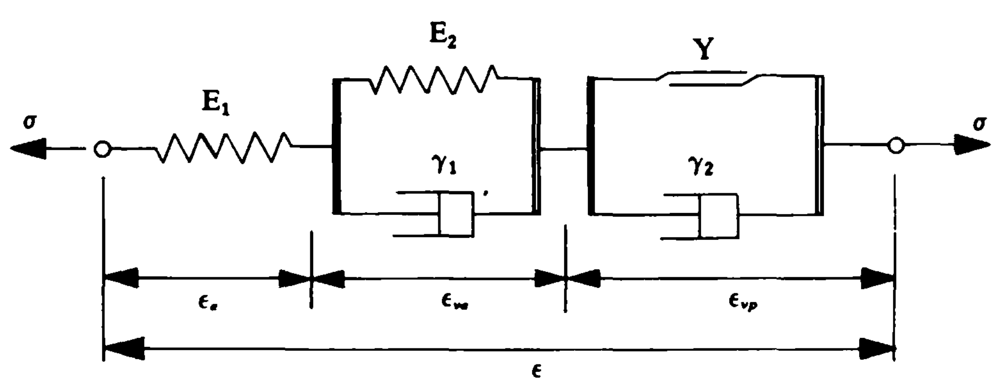


Fig. 4.9 Problem 4.2.

4.3 Use the unidimensional computer code developed in this chapter to determine the stress relaxation with time when the Maxwell model shown in Fig. 4.10 is subjected to a constant displacement condition. The critical time-step length for this model can be shown to be

$\Delta t = 2/\gamma E$. Solve the problem for several time-step lengths up to the critical value, thereby showing that numerical divergence occurs as soon as the limiting value is reached. For computation let $E = 100$, $\gamma = 0.01$ and $\phi_p = 0{\cdot}1$.

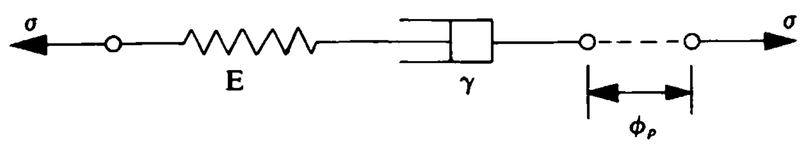


Fig. 4.10 Problem 4.3.

4.4 Modify the computer code developed in this chapter to allow solution of the material model of Problem 4.1.

4.5 In Section 4.9, Subroutine CONVP, monitoring convergence to steady state conditions, was based on a global criterion. Modify this subroutine so that convergence is based upon the condition

$$\frac{|\Delta\epsilon_{vp}{}^{n}|}{|\Delta\epsilon_{vp}{}^{1}|}\times 100 \leqslant \text{TOLER}, \tag{4.42}$$

for each individual element.

4.6 Develop the elastic stiffness matrix, $K^{(e)}$, for a two-node finite element in the form of a sphere and which is to be subjected to spherically symmetrical radial loading only. Assume a linear variation between nodes and note the following relationships

$$\epsilon_r = \frac{\partial u}{\partial r} = \frac{1}{E}[\sigma_r - \nu(\sigma_\theta + \sigma_\phi)]; \qquad \sigma_\theta = \sigma_\phi;$$

$$\epsilon_\theta = \epsilon_\phi = \frac{u}{r} = \frac{1}{E}[(1-\nu)\sigma_\theta - \nu\sigma_r], \tag{4.43}$$

in which u is the radial displacement and ϵ_r, ϵ_θ, ϵ_ϕ and σ_r, σ_θ, σ_ϕ are respectively the strain and stress components. Also express the stress components in terms of the nodal displacements.

4.7 Use the stiffness matrix evaluated in Problem 4.6 to modify the one-dimensional viscoplastic program UNVIS to allow solution of spherically symmetrical problems. Assume a Tresca yield criterion which implies commencement of yielding when $\sigma_r - \sigma_\theta = \sigma_Y$.

4.8 Employ the program developed in Problem 4.7 to determine the variation of the elasto-viscoplastic stress distribution with time in a sphere which is instantaneously loaded by an internal pressure of 500 N/mm². The internal and external radii of the sphere are 10 cm and 25 cm respectively, the elastic modulus $E = 2 \times 10^5$ N/mm^2, Poisson's ratio $\nu = 0{\cdot}3$, the uniaxial yield stress $\sigma_Y = 300$ N/mm^2, hardening paramater, $H' = 0$ and take the fluidity parameter $\gamma = 0{\cdot}001$. Compare your steady state solution with the theoretical elasto-plastic results of Ref. 2.

4.14 References

1. CORMEAU, I., Numerical stability in quasistatic elasto-visco-plasticity, *Int. J. Num. Meth. Engng.*, **9**, 109–127 (1975).
2. HILL, R., *The Mathematical Theory of Plasticity*, Oxford University Press, 1950.

Chapter 5
Elasto-plastic Timoshenko beam analysis

Written in collaboration with H. H. Abdel Rahman

5.1 Introduction

In this chapter we introduce some elasto-plastic beam formulations which are useful in their own right but which also provide insight into the elasto-plastic plate formulations presented later.

There are two main beam theories on which we could base our studies:

(i) *Euler–Bernoulli beam theory.* This theory, which is usually favoured by engineers because of its simplicity, takes no account of transverse shear deformation. The simplest Euler–Bernoulli beam element based on the displacement method is the well-known Hermitian element[1] with cubic displacements. Bending moments may vary linearly over this element.

(ii) *Timoshenko beam theory.* This theory allows for transverse shear deformation effects. The simplest Timoshenko beam element is the Hughes element[2] with linear displacements and normal rotations. Bending moments are constant over this element.

Although the Euler–Bernoulli theory is frequently adopted we choose the Timoshenko beam theory as a basis for our study of the elasto-plastic analysis of beams since we may make use of a finite element which involves constant bending moments and is more in keeping with the presentations given in the previous chapters. Furthermore, Timoshenko beam theory can rightly be considered as the one-dimensional precursor of Mindlin plate theory which is used in Chapter 9.

Firstly in this chapter the basic assumptions of Timoshenko beam theory are outlined. The Hughes element formulation is then presented for the elastic case.

There are two approaches to the elasto-plastic analysis of Timoshenko beams:

(i) *Non-layered approach.* In this method, when the bending moment reaches the yield moment, the whole cross-section of the beam is assumed to become plastic instantaneously. This is however a convenient fiction as in reality there is always a gradual plastification of the beam with the outer fibres becoming plastic initially. The zone of plasticification then spreads inwards until the whole section ultimately becomes plastic.

(ii) *Layered approach.* In this method we attempt to capture the spread of plasticity over the depth of the beam. The beam is thus divided into a number of layers each of which may become plastic separately. As the number of layers is increased, this model provides a more realistic representation of the gradual spread of plasticity over the beam cross-section.

Both non-layered and layered approaches are described in detail and program TIMOSH for the non-layered beams and program TIMLAY for the layered beams are presented and their use is illustrated with the aid of some examples.

5.2 The basic assumptions of Timoshenko beam theory

5.2.1 Introductory comments

There are several basic assumptions adopted in the derivation of the governing equations of Timoshenko beam theory. Here we reiterate these assumptions for elastic, small deflection analysis and then in later sections we present some extensions of the theory to allow for elasto-plastic analysis.

5.2.2 Assumed displacement field

In a typical Timoshenko beam, such as the one shown in Fig. 5.1, it is usual to assume that normals to the neutral axis before deformation remain straight but not necessarily normal to the neutral axis after deformation. This implies that the axial displacement $\bar{u}$ at any point (x, z) may be expressed directly in terms of $\theta(x)$ the rotation of the normal so that

$$\bar{u}(x, z) = -z\theta(x) \tag{5.1}$$

Note that the normal rotation $\theta(x)$ is equal to the slope of the neutral axis dw/dx minus a rotation β which is due to the transverse shear deformation.

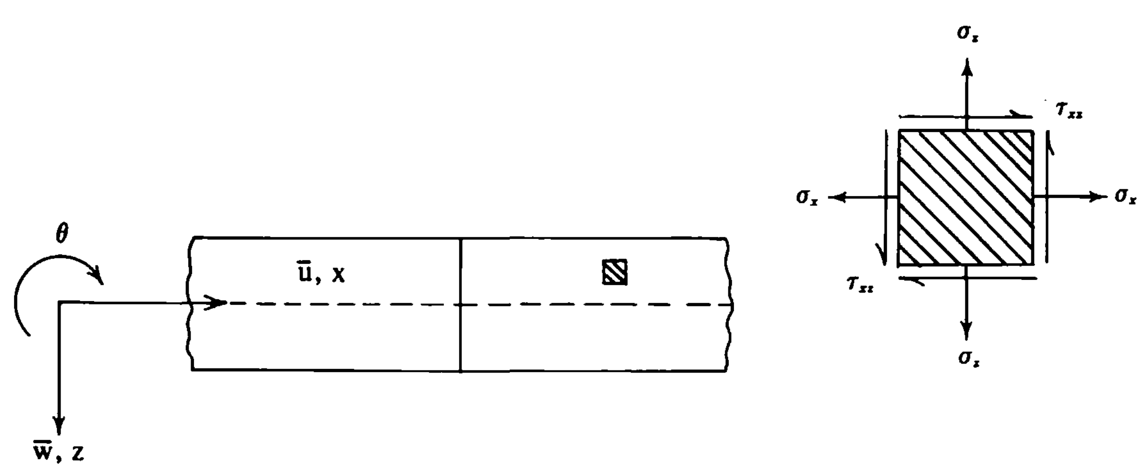


Fig. 5.1 Timoshenko beam.

Thus we have

$$\theta(x) = \frac{d\bar{w}}{dx} - \beta. \tag{5.2}$$

Notice also that the lateral displacement $\bar{w}$ at any point (x, z) is given by the lateral displacement at the neutral axis so that

$$\bar{w}(x, z) = w(x) \tag{5.3}$$

5.2.3 Stress–strain relationships

In Timoshenko beam theory, the elastic stress–strain relationships used for plane stress analysis are usually adopted in a slightly modified form. For convenience we assume that the beam is loaded in the xz plane and thus for an isotropic elastic material the relevant stress–strain relationships are

$$\begin{bmatrix} \sigma_x \\ \sigma_z \\ \tau_{xz} \end{bmatrix} = \frac{E}{(1-\nu^2)} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{(1-\nu)}{2} \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \epsilon_z \\ \gamma_{xz} \end{bmatrix} \tag{5.4}$$

where E is the Young's modulus and ν is the Poisson's ratio.

If σ_z is assumed to be equal to zero then

$$\epsilon_z = -\nu\epsilon_x \tag{5.5}$$

and by eliminating ϵ_z from (5.4) and (5.5), it is possible to write the following stress–strain relationship

$$\sigma_x = E\epsilon_x \quad \text{and} \quad \tau_{xz} = G\gamma_{xz} \tag{5.6}$$

where for an isotropic material $G = E/[2(1+\nu)]$ is the shear modulus.

5.2.4 Strain–displacement relationships

Usually small deflection theory is adopted and the axial strain ϵ_x is given as

$$\epsilon_x = \frac{\partial \bar{u}}{\partial x}. \tag{5.7}$$

If approximation (5.1) is adopted then this strain can be written as

$$\epsilon_x = -z\frac{d\theta}{dx}. \tag{5.8}$$

Similarly the shear strain γ_{xz} is given as

$$\gamma_{xz} = \frac{\partial \bar{u}}{\partial z} + \frac{\partial \bar{w}}{\partial x} \tag{5.9}$$

and if approximation (5.2) is adopted we obtain

$$\gamma_{xz} = -\theta + \frac{dw}{dx} = \beta. \qquad (5.10)$$

5.2.5 Virtual work expression

Consider a Timoshenko beam of depth t in which the breadth b varies with depth symmetrically about the neutral axis. The beam is subjected to a distributed loading of intensity q. If the beam undergoes a set of virtual lateral displacements δw, virtual normal rotations $\delta\theta$ and associated virtual curvatures $-z[d(\delta\theta)/dx]$ and virtual shear strains $\delta\beta$ then the virtual work equation can be written as

$$\int_0^l \int_{-t/2}^{t/2} \int_{b(-t/2)}^{b(t/2)} \left\{-z\frac{d(\delta\theta)}{dx}\sigma_x + \delta\beta\,\tau_{xz}\right\} dy\,dz\,dx - \int_0^l \delta w q\,dx = 0 \qquad (5.11)$$

or

$$\int_0^l \left(-\frac{d(\delta\theta)}{dx}M + \delta\beta\, Q\right)dx - \int_0^l \delta w q\,dx = 0$$

where the bending moment

$$M = \int_{-t/2}^{t/2} \int_{b(-t/2)}^{b(t/2)} z\,\sigma_x\,dy\,dz \qquad (5.12)$$

and the shear force

$$Q = \int_{-t/2}^{t/2} \int_{b(-t/2)}^{b(t/2)} \tau_{xz}\,dy\,dz. \qquad (5.13)$$

Using (5.12) and (5.13), if we substitute for σ_x and τ_{xz} in (5.6) respectively we obtain

$$M = \left(\int_{-t/2}^{t/2} \int_{b(-t/2)}^{b(t/2)} z^2 E\,dy\,dz\right)\left(-\frac{d\theta}{dx}\right) = EI\left(-\frac{d\theta}{dx}\right) \qquad (5.14)$$

and

$$Q = \left(\int_{-t/2}^{t/2} \int_{b(-t/2)}^{b(t/2)} G\,dy\,dz\right)(\beta) = GA\,\beta \qquad (5.15)$$

where EI is the flexural rigidity and GA, the shear rigidity, is replaced by $G\hat{A}$ where the area A is replaced by A/α. The parameter α is a correction factor to allow for cross-sectional warping. For a rectangular section α is usually taken as 1·5.*

* Many different definitions of α have been presented in the various papers on Timoshenko beams. Cowper[3] summarises some definitions for beams of various cross-sections. For example, he shows that α may be taken as $(12+11\nu)/(10+10\nu)$ for rectangular cross-sections and $(7+6\nu)/(6+6\nu)$ for circular cross-sections. Here we take $\alpha = 1\cdot5$ unless otherwise stated.

If we substitute for M and Q from (5.14) and (5.15) we can rewrite the virtual work equation (5.11) as

$$\int_0^l \left(\frac{d(\delta\theta)}{dx} EI \frac{d\theta}{dx} + \delta\beta\, G\hat{A}\, \beta - \delta\, wq\right) dx = 0 \tag{5.16}$$

5.2.6 A comparison of various beam approximations

In order to compare the various beam approximations consider a simply supported beam of rectangular cross-section, flexural rigidity EI, Poisson's ratio ν, depth t and length L which is subjected to a uniformly distributed loading q. The lateral deflection in the elastic range is given as

(i) $$w = \frac{qL^4}{24EI}\left\{\left[\left(\frac{x}{L}\right)^4 - \frac{3}{2}\left(\frac{x}{L}\right)^2 + \frac{5}{16}\right] + \left(\frac{t}{L}\right)^2\left[\frac{12}{5} + \frac{3\nu}{2}\right]\left[\frac{1}{4} - \left(\frac{x}{L}\right)^2\right]\right\} \tag{5.17a}$$

when plane stress (PS) assumptions are adopted,

(ii) $$w = \frac{qL^4}{24EI}\left\{\left[\left(\frac{x}{L}\right)^4 - \frac{3}{2}\left(\frac{x}{L}\right)^2 + \frac{5}{16}\right] + \left(\frac{t}{L}\right)^2 [2\alpha(1+\nu)]\left[\frac{1}{4} - \left(\frac{x}{L}\right)^2\right]\right\} \tag{5.17b}$$

when Timoshenko beam (TB) assumptions are adopted and

(iii) $$w = \frac{qL^4}{24EI}\left\{\left[\left(\frac{x}{L}\right)^4 - \frac{3}{2}\left(\frac{x}{L}\right)^2 + \frac{5}{16}\right]\right\} \tag{5.17c}$$

when Euler–Bernoulli (EB) assumptions are adopted.

Thus, for long slender beams in which (t/L) is small, EB theory is adequate If we take Cowper's value [3] of $\alpha = (12+11\nu)/(10+10\nu)$ then the ratio of the second-order additional lateral deflections due to shear deformation obtained under TB and PS assumptions is $(24+22\nu)/(24+15\nu)$ which varies from 1·00 to 1·11 as ν varies from 0·0 to 0·5. Thus TB theory is an accurate theory for beams of all dimensions.

5.3 Finite element idealisation for linear elastic Timoshenko beams

5.3.1 Introduction

The theoretical and programming aspects of the finite element analysis of linear elastic Timoshenko beams have been dealt with in detail in previous books by the authors[1, 5]. Here we derive the stiffness matrix and consistent load vector for a linear element and set the scene for the analysis of elasto-plastic Timoshenko beams which will be discussed later.

5.3.2 Displacement and strain representation

In the Hughes element representation, the lateral displacement w is represented by the relationship

$$w^{(e)} = N_1^{(e)} w_1^{(e)} + N_2^{(e)} w_2^{(e)} \tag{5.18}$$

where $w_1^{(e)}$ and $w_2^{(e)}$ are the nodal lateral displacements at local nodes 1 and 2 of element e and the shape functions (shown in Fig. 5.2) are

$$N_1^{(e)} = (x_2^{(e)} - x^{(e)})/l^{(e)}$$

and

$$N_2^{(e)} = (x^{(e)} - x_1^{(e)})/l^{(e)}$$

in which $x_1^{(e)}$ and $x_2^{(e)}$ are the x-coordinates of local nodes 1 and 2, $x^{(e)}$ is the x-coordinate of a point within the element and $l^{(e)}$ is the length of the element.

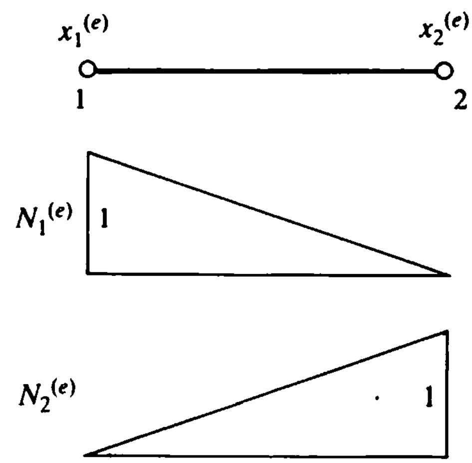


Fig. 5.2 Beam element shape functions.

Similarly the normal rotation $\theta^{(e)}$ within element e is represented as

$$\theta^{(e)} = N_1^{(e)} \theta_1^{(e)} + N_2^{(e)} \theta_2^{(e)} \tag{5.19}$$

where $\theta_1^{(e)}$ and $\theta_2^{(e)}$ are the normal rotations at local nodes 1 and 2 of element e.

The curvature–displacement relationship can be expressed as

$$-\left(\frac{d\theta}{dx}\right)^{(e)} = -\left(\frac{dN_1}{dx}\right)^{(e)} \theta_1^{(e)} - \left(\frac{dN_2}{dx}\right)^{(e)} \theta_2^{(e)} \tag{5.20}$$

or

$$\epsilon_f^{(e)} = \left[0,\ \frac{1}{l^{(e)}},\ 0,\ -\frac{1}{l^{(e)}}\right] \begin{bmatrix} w_1^{(e)} \\ \theta_1^{(e)} \\ w_2^{(e)} \\ \theta_2^{(e)} \end{bmatrix} = \boldsymbol{B}_f^{(e)} \boldsymbol{\varphi}^{(e)}$$

where $\boldsymbol{B}_f^{(e)}$ is the curvature–displacement matrix.

The shear strain–displacement relationship is given as

$$\left(\frac{dw}{dx}-\theta\right)^{(e)}=\left(\frac{dN_1}{dx}\right)^{(e)}w_1{}^{(e)}-N_1{}^{(e)}\,\theta_1{}^{(e)}+\left(\frac{dN_2}{dx}\right)^{(e)}w_2{}^{(e)}-N_2{}^{(e)}\,\theta_2{}^{(e)} \quad (5.21)$$

or

$$\epsilon_s{}^{(e)}=\left[-\frac{1}{l^{(e)}},\ -\frac{(x_2{}^{(e)}-x^{(e)})}{l^{(e)}},\ \frac{1}{l^{(e)}},\ -\frac{(x^{(e)}-x_1{}^{(e)})}{l^{(e)}}\right]\begin{bmatrix} w_1{}^{(e)} \\ \theta_1{}^{(e)} \\ w_2{}^{(e)} \\ \theta_2{}^{(e)} \end{bmatrix}=\boldsymbol{B}_s{}^{(e)}\,\boldsymbol{\varphi}^{(e)}$$

where $\boldsymbol{B}_s{}^{(e)}$ is the shear strain–displacement matrix.

5.3.3 Stiffness matrix evaluation

Given the element strain–displacement relationships outlined in Section 5.3.2, Hughes has shown that using a virtual work approach the governing equations can be expressed as

$$[\boldsymbol{K}_f+\boldsymbol{K}_s]\boldsymbol{\varphi}-\boldsymbol{f}=0 \quad (5.22)$$

where the submatrices of $\boldsymbol{K}_f$ and $\boldsymbol{K}_s$ and subvectors of $\boldsymbol{f}$ for element e can be written as

$$\boldsymbol{K}_f{}^{(e)}=\int_{x_1{}^{(e)}}^{x_2{}^{(e)}}[\boldsymbol{B}_f{}^{(e)}]^T(EI)^{(e)}\,\boldsymbol{B}_f{}^{(e)}\,dx$$

$$\boldsymbol{K}_s{}^{(e)}=\int_{x_1{}^{(e)}}^{x_2{}^{(e)}}[\boldsymbol{B}_s{}^{(e)}]^T(G\hat{A})^{(e)}\,\boldsymbol{B}_s{}^{(e)}\,dx$$

$$\boldsymbol{f}^{(e)}=\int_{x_1{}^{(e)}}^{x_2{}^{(e)}}[N_1{}^{(e)},\ 0,\ N_2{}^{(e)},\ 0]^T q\,dx. \quad (5.23)$$

The flexural element stiffness matrix can be evaluated using a 1-point Gauss–Legendre rule and takes the form

$$\boldsymbol{K}_f{}^{(e)}=\left(\frac{EI}{l}\right)^{(e)}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \quad (5.24)$$

If $\boldsymbol{K}_s{}^{(e)}$ is evaluated exactly using a 2-point Gauss–Legendre rule we obtain

$$K_s^{(e)} = \left(\frac{G\hat{A}}{l}\right)^{(e)} \begin{bmatrix} 1 & \frac{l}{2} & -1 & \frac{l}{2} \\ \frac{l}{2} & \frac{l^2}{3} & -\frac{l}{2} & \frac{l^2}{6} \\ -1 & -\frac{l}{2} & 1 & -\frac{l}{2} \\ \frac{l}{2} & \frac{l^2}{6} & -\frac{l}{2} & \frac{l^2}{3} \end{bmatrix}^{(e)} \tag{5.25}$$

Unfortunately it has been shown that with this formulation, overstiff solutions are obtained. This phenomenon, known as locking, may be 'cured' by integrating $K_s^{(e)}$ with a 1-point Gauss–Legendre rule. If such a selectively integrated element is adopted we find that

$$K_s^{(e)} = \left(\frac{G\hat{A}}{l}\right)^{(e)} \begin{bmatrix} 1 & \frac{l}{2} & -1 & \frac{l}{2} \\ \frac{l}{2} & \frac{l^2}{4} & -\frac{l}{2} & \frac{l^2}{4} \\ -1 & -\frac{l}{2} & 1 & -\frac{l}{2} \\ \frac{l}{2} & \frac{l^2}{4} & -\frac{l}{2} & \frac{l^2}{4} \end{bmatrix}^{(e)} \tag{5.26}$$

and the results obtained are excellent.

The consistent nodal force vector is given as

$$f^{(e)} = \left[\frac{(ql)^{(e)}}{2},\ 0,\ \frac{(ql)^{(e)}}{2},\ 0\right] \tag{5.27}$$

which, unlike the Euler–Bernōulli cubic Hermitian element, only has lateral nodal point forces.

For the nonlayered elasto-plastic Timoshenko beam finite element analysis, when the beam bending moment reaches the yield moment M_0, the whole element becomes plastic and acts as a plastic hinge. In such a situation the flexural rigidity EI is replaced by an elasto-plastic flexural rigidity $(EI)_{ep}$ whereas the shear rigidity $G\hat{A}$ is assumed to be unchanged.

5.3.4 Element stress resultants

We can obtain expressions which enable us to calculate the bending moments and shear forces within each element using (5.14) and (5.15). The bending moment, which is constant in each element e, is given as

$$M^{(e)} = (EI)^{(e)} \boldsymbol{B}_f{}^{(e)} \boldsymbol{\varphi}^{(e)} = (EI)^{(e)} \left[0, \ \frac{1}{l^{(e)}}, \ 0, \ -\frac{1}{l^{(e)}}\right] \begin{bmatrix} w_1{}^{(e)} \\ \theta_1{}^{(e)} \\ w_2{}^{(e)} \\ \theta_2{}^{(e)} \end{bmatrix}$$

$$= \left(\frac{EI}{l}\right)^{(e)} (\theta_1{}^{(e)} - \theta_2{}^{(e)}). \tag{5.28}$$

The shear force varies linearly over each element but we evaluate it at

$$x = \frac{x_1{}^{(e)} + x_2{}^{(e)}}{2}$$

and assume it to be constant over the element. This is consistent with the practice of using selective integration in the evaluation of $\boldsymbol{K}^{(e)}$. The shear force is therefore given as

$$Q^{(e)} = (G\hat{A})^{(e)} \boldsymbol{B}_s{}^{(e)} \boldsymbol{\varphi}^{(e)} = (G\hat{A})^{(e)} \left[-\frac{1}{l^{(e)}}, \ -\frac{1}{2}, \ \frac{1}{l^{(e)}}, \ -\frac{1}{2}\right] \begin{bmatrix} w_1{}^{(e)} \\ \theta_1{}^{(e)} \\ w_2{}^{(e)} \\ \theta_2{}^{(e)} \end{bmatrix}$$

$$= (G\hat{A})^{(e)} \left\{ \left(\frac{w_2{}^{(e)} - w_1{}^{(e)}}{l^{(e)}} \right) - \left(\frac{\theta_1{}^{(e)} + \theta_2{}^{(e)}}{2} \right) \right\}. \tag{5.29}$$

5.4 Elasto-plastic nonlayered Timoshenko beams

5.4.1 The yield moment

Consider a Timoshenko beam subjected to a bending moment. Timoshenko's assumptions imply that the axial stress and strain vary linearly across the depth of the section. As the bending moment is increased the yield stress is attained at the top and bottom fibres and with a further increase the yield will spread from these outer fibres inwards until the two zones of yield meet. The cross-section is then said to be fully plastic. It should be noted that the interaction of σ_x and τ_{xz} has been ignored during yield. This is inexact, but experience shows that the effect is not of prime importance especially when thin beams are considered.

The value of this ultimate moment in the fully plastic condition can be calculated in terms of the yield stress σ_0.* Thus

$$M_0 = \int_{b(-t/2)}^{b(t/2)} \int_{-t/2}^{t/2} z\, \sigma_0 \, dz \, dy \tag{5.30}$$

* Note that for beam and plate problems the uniaxial yield stress is designated by σ_0 and not σ_Y.

and for a rectangular beam of breadth b, $M_0 = \sigma_0(bt^2/4)$. However, it should be noted that the assumption used in the finite element solution implies that the whole cross-section becomes plastic as soon as the bending moment reaches its yield value M_0. This means that, for the beam case shown in Fig. 5.3, the whole cross-section is assumed to be plastic when the bending moment of situation (c) becomes equal to the bending moment of situation (d)—in which case the extreme fibre stress in situation (c) exceeds the actual yield stress of the material.

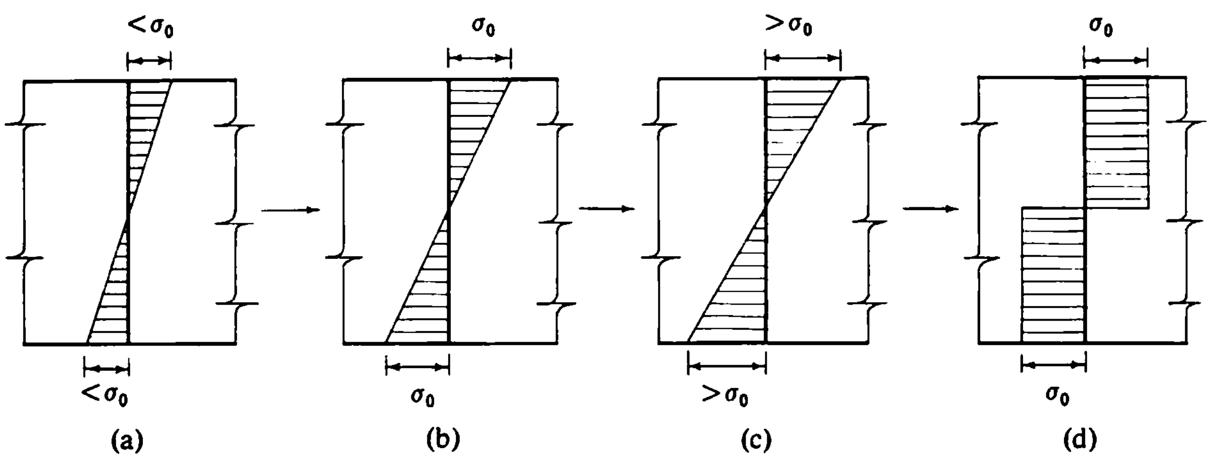


Fig. 5.3 Yielding of non-layered beam.

5.4.2 Elasto-plastic bending

As mentioned earlier, elasto-plastic behaviour is characterised by an initial elastic material response with an additional plastic deformation when the bending moment $|M|$ exceeds the yield moment M_0. The plastic deformation is irreversible on unloading and its onset is governed by a very simple yield criterion. Post-yield deformation usually occurs with a considerably reduced material stiffness.

The moment–curvature relationship for a Timoshenko beam of elasto-plastic material is shown in Fig. 5.4. The beam initially deforms elastically with a flexural rigidity of EI until the ultimate bending moment is reached at which stage the whole beam cross-section becomes plastic. On increasing the load further, the material is assumed to exhibit linear strain-hardening characterised by the tangential flexural rigidity $(EI)_T$.

At some stage after initial yielding consider a further load application resulting in an incremental increase of bending moment accompanied by a change of curvature $d\epsilon_f$. Assuming that the curvature can be separated into elastic and plastic components, so that

$$d\epsilon_f = (d\epsilon_f)_e + (d\epsilon_f)_p, \tag{5.31}$$

we define as a strain hardening parameter

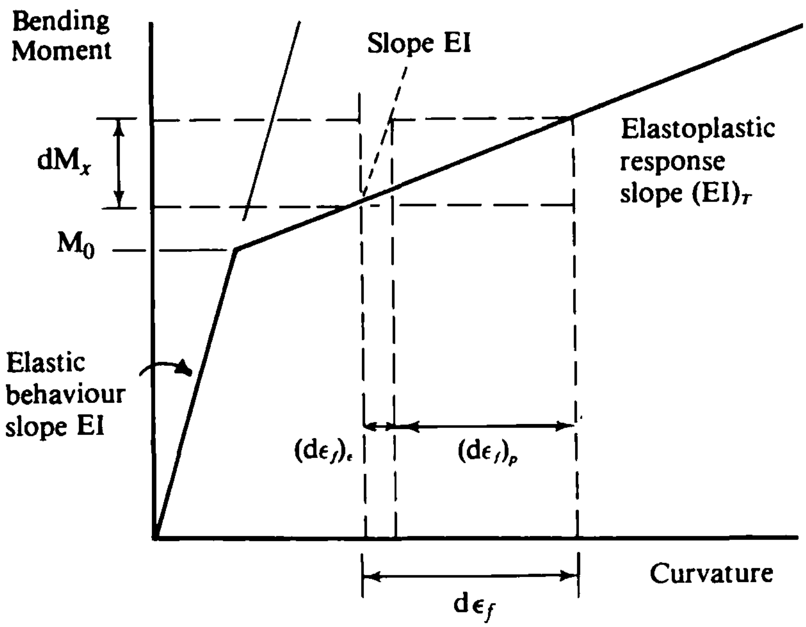


Fig. 5.4 Moment curvature relationship for a Timoshenko beam.

$$H' = \frac{dM}{(d\epsilon_f)_p}.$$

This can be interpreted as the slope of the strain-hardening portion of the moment–curvature curve after the removal of the elastic curvature component. Thus

$$H' = \frac{dM}{d\epsilon_f - (d\epsilon_f)_e} = \frac{(EI)_T}{1 - [(EI)_T/EI]}. \tag{5.32}$$

It is therefore possible to rewrite (5.31) as

$$d\epsilon_f = \frac{dM}{EI} + \frac{dM}{H'} = \frac{dM(H' + EI)}{EIH'} \tag{5.33}$$

and then the incremental moment–curvature relationship can be written in the form

$$dM = \frac{EIH'}{(EI + H')} d\epsilon_f. \tag{5.34}$$

Thus during yielding the incremental stress–strain resultant relationship is

$$dM = EI\left(1 - \frac{EI}{EI + H'}\right) d\epsilon_f$$

$$dQ = G\hat{A}\, d\epsilon_s. \tag{5.35}$$

The shear force/shear strain relationship is always elastic whereas the moment–curvature relationship is elasto-plastic. After yielding the flexural rigidity EI is replaced by

$$EI\left(1-\frac{EI}{EI+H'}\right).$$

If the hardening parameter H' is equal to zero then the material behaviour is elasto-perfectly plastic and as mentioned in Section 3.5 for elasto-plastic axial bar elements this may lead to tangential stiffness matrices which are singular. This difficulty can also be avoided by use of the initial stiffness method in which the elastic element stiffnesses are employed at every stage of the computation thereby guaranteeing a positive definite assembled stiffness matrix.

5.4.3 Solution of nonlinear equations

Let us now generate the nonlinear equilibrium equations using the virtual expression (5.11). In order to do this we require the global rather than the element expressions for the lateral displacements, rotation, curvature and shear strain. At any point in the finite element mesh the lateral displacement and rotation can be obtained from the expression

$$\begin{bmatrix} w \\ \theta \end{bmatrix} = \boldsymbol{N}\boldsymbol{\varphi} \tag{5.36}$$

where the shape function matrix is

$$\boldsymbol{N} = \begin{bmatrix} N_1, & 0, & N_2, & 0, & \ldots, & N_n, & 0 \\ 0, & N_1, & 0, & N_2, & \ldots, & 0, & N_n \end{bmatrix} \tag{5.37}$$

and the vector of nodal displacements is

$$\boldsymbol{\varphi} = [w_1, \theta_1, w_2, \theta_2, \ldots, w_n, \theta_n]^T \tag{5.38}$$

where w_i, θ_i and N_i are the lateral displacement, rotation and global shape functions associated with node i.

The curvature and shear strain at any point within the entire finite element mesh is given as

$$-\frac{d\theta}{dx} = \boldsymbol{B}_f\boldsymbol{\varphi} \quad \text{and} \quad \frac{dw}{dx}-\theta = \boldsymbol{B}_s\boldsymbol{\varphi} \tag{5.39}$$

where

$$\boldsymbol{B}_f = \left[0, \; -\frac{dN_1}{dx}, \; 0, \; -\frac{dN_2}{dx}, \; \ldots, \; 0, \; -\frac{dN_n}{dx}\right] \tag{5.40}$$

and

$$\boldsymbol{B}_s = \left[\frac{dN_1}{dx}, \; -N_1, \; \frac{dN_2}{dx}, \; -N_2, \; \ldots, \; \frac{dN_n}{dx}, \; -N_n\right] \tag{5.41}$$

Virtual curvatures and shear strains are given as

$$-\frac{d(\delta\theta)}{dx} = \boldsymbol{B}_f\,\delta\boldsymbol{\varphi} \quad \text{and} \quad \frac{d(\delta w)}{dx} - \delta\theta = \boldsymbol{B}_s\,\delta\boldsymbol{\varphi} \tag{5.42}$$

respectively, where the vector of virtual nodal displacements is written as

$$\delta\boldsymbol{\varphi} = [\delta w_1, \delta\theta_1, \delta w_2, \delta\theta_2, \ldots, \delta w_n, \delta\theta_n]^T. \tag{5.43}$$

Thus the virtual work expression (5.11) can now be written as

$$\int_0^l [\delta\boldsymbol{\varphi}]^T [\boldsymbol{B}_f]^T M\,dx + \int_0^l [\delta\boldsymbol{\varphi}]^T [\boldsymbol{B}_s]^T Q\,dx - \int_0^l [\delta\boldsymbol{\varphi}]^T [\bar{\mathbf{N}}]^T q\,dx = 0 \tag{5.44}$$

where

$$\bar{\mathbf{N}} = [N_1, 0, N_2, 0, \ldots, N_n, 0]. \tag{5.45}$$

Since (5.44) must be true for any set of virtual displacements $\delta\boldsymbol{\varphi}$ then we have

$$\left\{\int_0^l [\boldsymbol{B}_f]^T M\,dx + \int_0^l [\boldsymbol{B}_s]^T Q\,dx\right\} - \int_0^l [\bar{\mathbf{N}}]^T q\,dx = 0 \tag{5.46}$$

or

$$\boldsymbol{p} - \boldsymbol{f} = 0.$$

In fact this equation is identical to (5.22) when there is no plasticity.

Unfortunately in elasto-plastic problems M is a nonlinear function and in general we can only predict the vector $\boldsymbol{p}$ approximately. Thus (5.46) is nonlinear and since $\boldsymbol{p}$ is only approximately known than $\boldsymbol{p}-\boldsymbol{f}$ will equal a residual value $\boldsymbol{\psi}(\boldsymbol{\varphi})$ which we attempt to reduce to zero in our solution procedure.

We evaluate contributions to $\boldsymbol{p}$ element by element and assemble in the usual manner. The contribution from element e has the form

$$\boldsymbol{p}^{(e)} = \int_{x_1^{(e)}}^{x_2^{(e)}} \begin{bmatrix} 0 \\ \dfrac{1}{l^{(e)}} \\ 0 \\ -\dfrac{1}{l^{(e)}} \end{bmatrix} M^{(e)}\,dx + \int_{x_1^{(e)}}^{x_2^{(e)}} \begin{bmatrix} -\dfrac{1}{l^{(e)}} \\ \dfrac{x^{(e)} - x_2^{(e)}}{l^{(e)}} \\ \dfrac{1}{l^{(e)}} \\ \dfrac{x_1^{(e)} - x^{(e)}}{l^{(e)}} \end{bmatrix} Q^{(e)}\,dx$$

$$= \left[-Q^{(e)},\ M^{(e)} - \frac{(Ql)^{(e)}}{2},\ Q^{(e)},\ -M^{(e)} - \frac{(Ql)^{(e)}}{2}\right]^T. \tag{5.47*}$$

*The second integral evaluation is equivalent to using a 1-point Gauss rule.

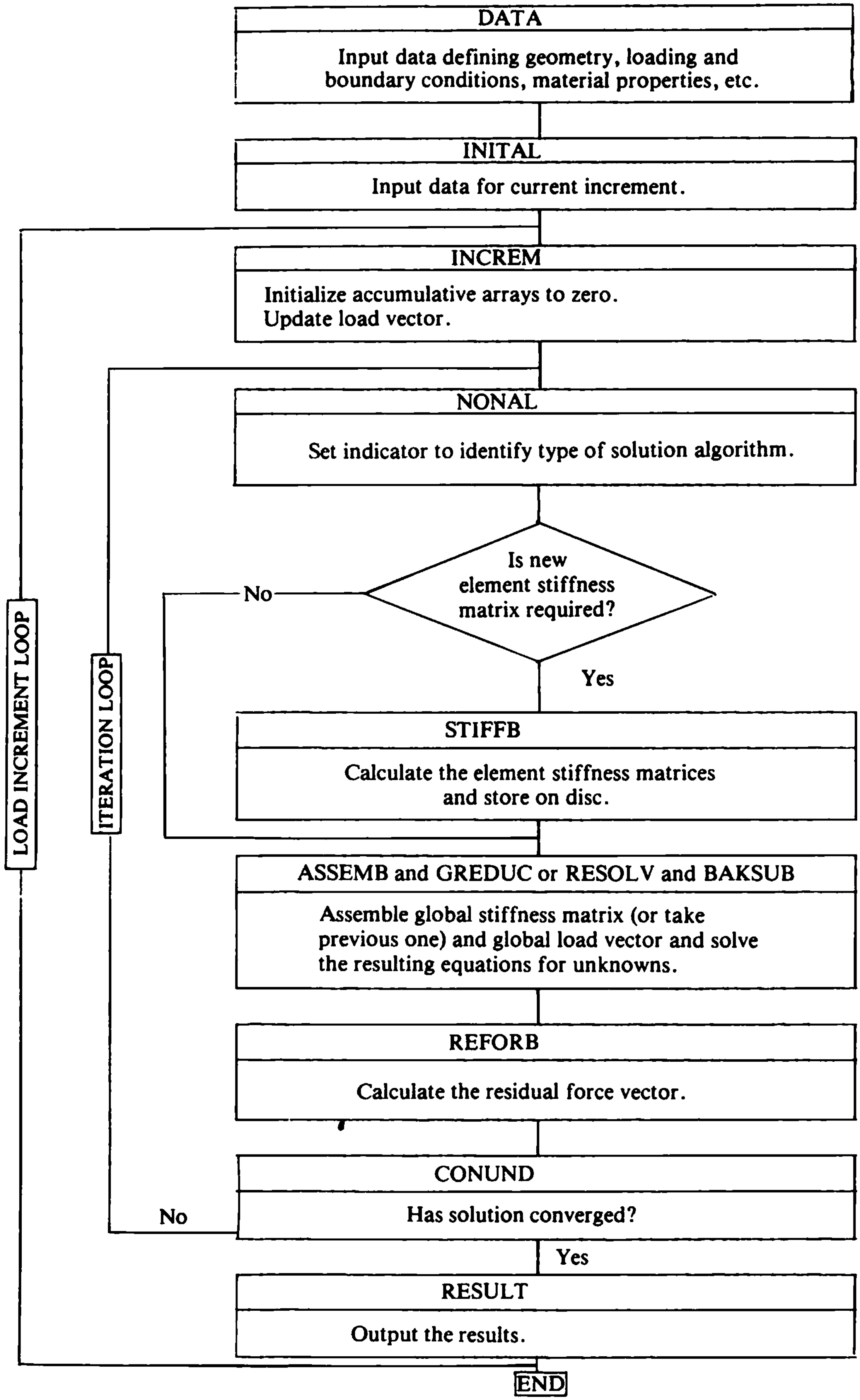


Fig. 5.5 Overall structure of program TIMOSH.

Note that the appropriate value of bending moment $M^{(e)}$ is inserted in (5.47).

Table 5.1 shows the complete sequence of nonlinear equation solving which is very similar to the one adopted for the axially-loaded bars in Chapter 3.

1. Begin load increment.
   Set $\mathbf{f} = \mathbf{f}+\Delta\mathbf{f}$, iteration counter $i = 0$ and $\psi^i = \Delta\mathbf{f}+\psi$ (that is, include equilibrium correction from previous increment).
2. Evaluate the new tangential stiffness matrix $\mathbf{K}_T$, if necessary.
3. Solve $\psi^i = \mathbf{K}_T \Delta\varphi^i$
4. Evaluate $\varphi = \varphi + \Delta\varphi^i$.
5. For each element evaluate $M^{(e)}$ and $Q^{(e)}$. Check $M^{(e)}$ and adjust its value accordingly to account for any plastic behaviour. Evaluate the element residual force vector $[\psi^{(e)}]^{i+1} = \mathbf{p}^{(e)} - \mathbf{f}^{(e)}$ and assemble into the global residual force vector ψ^{i+1}.
6. Check $\Delta\varphi^i$ for convergence.
7. If solution has converged set $\psi = \psi^{i+1}$ and go to step 1, otherwise set $i = i+1$ and go to step 2.

Table 5.1 Solution procedure for elasto-plastic nonlayered Timoshenko beam analysis.

5.4.4 Overall program structure of TIMOSH

A modular approach is adopted for program TIMOSH. In fact the overall structure is identical to the structure in the programs of Chapter 3. Figure 5.5 shows the overall structure of TIMOSH. Routines DATA, INITAL, INCREM, NONAL, ASSEMB, GREDUC, BAKSUB, CONUND, RESOLV and RESULT have already been described in Chapter 3. The only new routines are STIFFB, REFORB and, of course, the MASTER routine BEAM.

5.4.5 New routines for nonlayered elasto-plastic Timoshenko beam analysis

Master BEAM The master calling routine BEAM simply organises the calling of the main routines as described in Fig. 5.5.

```
      MASTER BEAM                                                       EPBM
C***********************************************************************EPBM
C                                                                       EPBM
C *** ELSTO-PLASTIC NONLAYERED TIMOSHENKO BEAM PROGRAM                  EPBM
C                                                                       EPBM
C***********************************************************************EPBM
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,     EPBM
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     EPBM
     .             NITER,NOUTP,FACTO                                    EPBM
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),          EPBM
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       EPBM
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),           EPBM
```

```
     .            TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .            REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      CALL  DATA
      CALL  INITAL
      DO 30 IINCS=1,NINCS
      CALL  INCLOD
      DO 10 IITER=1,NITER
      CALL  NONAL
      IF(KRESL.EQ.1) CALL  STIFFB
      CALL  ASSEMB
      IF(KRESL.EQ.1) CALL  GREDUC
      IF(KRESL.EQ.2) CALL  RESOLV
      CALL  BAKSUB
      CALL  REFORB
      CALL  CONUND
      IF(NCHEK.EQ.0) GO TO 20
      IF(IITER.EQ.1.AND.NOUTP.EQ.1) CALL  RESULT
      IF(NOUTP.EQ.2) CALL  RESULT
   10 CONTINUE
      WRITE(6,900)
  900 FORMAT(1H0,5X,'SOLUTION NOT CONVERGED')
      STOP
   20 CALL  RESULT
   30 CONTINUE
      STOP
      END
```

Subroutine STIFFB The purpose of this routine is to evaluate the element stiffness matrices and store them on disc prior to their use in the assembly and equation solving routines.

```
      SUBROUTINE  STIFFB
C**********************************************************************
C
C *** CALCULATES ELEMENT STIFFNESS MATRICES
C
C**********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .            KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .            NITER,NOUTP,FACTO
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .            FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .            MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .            TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .            REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      REWIND 1
      DO 20 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      EIVAL=PROPS(LPROP,1)
      SVALU=PROPS(LPROP,2)
      HARDS=PROPS(LPROP,4)
      NODE1=LNODS(IELEM,1)
      NODE2=LNODS(IELEM,2)
      ELENG=ABS(COORD(NODE2)-COORD(NODE1))
      IF(PLAST(IELEM).NE.0.0) EIVAL=EIVAL*(1.0-EIVAL/(EIVAL+HARDS))
      VALU1=0.5*SVALU
      VALU2=SVALU/ELENG
      VALU3=EIVAL/ELENG
      VALU4=0.25*SVALU*ELENG
      ESTIF(1,1)= VALU2
      ESTIF(1,2)= VALU1
```

```
      ESTIF(1,3)=-VALU2                                                 STFB  31
      ESTIF(1,4)= VALU1                                                 STFB  32
      ESTIF(2,2)= VALU3+VALU4                                           STFB  33
      ESTIF(2,3)=-VALU1                                                 STFB  34
      ESTIF(2,4)=-VALU3+VALU4                                           STFB  35
      ESTIF(3,3)= VALU2                                                 STFB  36
      ESTIF(3,4)=-VALU1                                                 STFB  37
      ESTIF(4,4)= VALU3+VALU4                                           STFB  38
      DO 10 ISTIF=1,4                                                   STFB  39
      DO 10 JSTIF=ISTIF,4                                               STFB  40
   10 ESTIF(JSTIF,ISTIF)=ESTIF(ISTIF,JSTIF)                             STFB  41
      WRITE(1) ESTIF                                                    STFB  42
   20 CONTINUE                                                          STFB  43
      RETURN                                                            STFB  44
      END                                                               STFB  45
```

STFB 15	Rewind disc ready for writing element stiffnesses.
STFB 16–38	For each element evaluate the upper triangular portion of the element stiffness matrix $K^{(e)}$. Note that if plasticity has taken place the elastic EI is replaced by the elasto-plastic $(EI)_T$.
STFB 39–41	Obtain using symmetry the lower triangular portion of $K^{(e)}$.
STFB 42	Write all element stiffness matrices on to disc.

Subroutine REFORB This routine evaluates the equivalent nodal forces.

```
      SUBROUTINE REFORB
C***********************************************************************
C
C *** CALCULATES INTERNAL EQUIVALENT NODAL FORCES
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLOAD,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO
      COMMON/UNIM2/PROPS(5,4),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(25),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4)
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=0.0
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      EIVAL=PROPS(LPROP,1)
      SVALU=PROPS(LPROP,2)
      YIELD=PROPS(LPROP,3)
      HARDS=PROPS(LPROP,4)
      NODE1=LNODS(IELEM,1)
      NODE2=LNODS(IELEM,2)
      ELENG=ABS(COORD(NODE2)-COORD(NODE1))
      WNOD1=XDISP(NODE1*NDOFN-1)
      WNOD2=XDISP(NODE2*NDOFN-1)
      THTA1=XDISP(NODE1*NDOFN)
      THTA2=XDISP(NODE2*NDOFN)
      STRAN=(THTA1-THTA2)/ELENG
      STLIN=STRAN*EIVAL
      STCUR=STRES(IELEM,1)+STLIN
      PREYS=YIELD+HARDS*ABS(PLAST(IELEM))
      IF(ABS(STRES(IELEM,1)).GE.PREYS) GO TO 20
```

```
      ESCUR=ABS(STCUR)-PREYS                                            RFRB  36
      IF(ESCUR.LE.0.0) GO TO 40                                         RFRB  37
      RFACT=ESCUR/ABS(STLIN)                                            RFRB  38
      GO TO 30                                                          RFRB  39
   20 IF(STRES(IELEM,1).GT.0.0.AND.STLIN.LE.0.0) GO TO 40               RFRB  40
      IF(STRES(IELEM,1).LT.0.0.AND.STLIN.GE.0.0) GO TO 40               RFRB  41
      RFACT=1.0                                                         RFRB  42
   30 REDUC=1.0-RFACT                                                   RFRB  43
      STRES(IELEM,1)=STRES(IELEM,1)+REDUC*STLIN+                        RFRB  44
     .               RFACT*EIVAL*(1.0-EIVAL/(EIVAL+HARDS))*STRAN        RFRB  45
      PLAST(IELEM)=PLAST(IELEM)+RFACT*STRAN*EIVAL/(EIVAL+HARDS)         RFRB  46
      GO TO 50                                                          RFRB  47
   40 STRES(IELEM,1)=STRES(IELEM,1)+STLIN                               RFRB  48
   50 STRES(IELEM,2)=STRES(IELEM,2)+(SVALU/ELENG)*(WNOD2-WNOD1)         RFRB  49
     .               -0.5*SVALU*(THTA1+THTA2)                           RFRB  50
      ELOAD(IELEM,1)=ELOAD(IELEM,1)-STRES(IELEM,2)                      RFRB  51
      ELOAD(IELEM,2)=ELOAD(IELEM,2)+STRES(IELEM,1)                      RFRB  52
     .               -0.5*ELENG*STRES(IELEM,2)                          RFRB  53
      ELOAD(IELEM,3)=ELOAD(IELEM,3)+STRES(IELEM,2)                      RFRB  54
      ELOAD(IELEM,4)=ELOAD(IELEM,4)-STRES(IELEM,1)                      RFRB  55
     .               -0.5*ELENG*STRES(IELEM,2)                          RFRB  56
   70 CONTINUE                                                          RFRB  57
      RETURN                                                            RFRB  58
      END                                                               RFRB  59
```

RFRB 15–17 Zero space for storing $\boldsymbol{p}$.
RFRB 18–57 For each element evaluate $\boldsymbol{p}^{(e)}$ and assemble into $\boldsymbol{p}$.

5.4.6 Examples of nonlayered elasto-plastic Timoshenko beam analysis

Two numerical examples are considered. The first example, Example 5.1, involves the yielding of a rectangular simple beam under uniformly distributed load. The beam material has the following properties:

$$E = 210{\cdot}0 \text{ kN/mm}^2$$
$$\nu = 0{\cdot}3$$
$$\sigma_0 = 0{\cdot}25 \text{ kN/mm}^2$$
$$H' = 0{\cdot}0$$

and the beam proportions are:

$$b = 150 \text{ mm}$$
$$t = 300 \text{ mm}$$
$$l = 3000 \text{ mm}$$

Typical input data is provided in Appendix IV.

The problem, finite element idealisation employed and the results are illustrated in Fig. 5.6, which shows that the beam fails as soon as a plastic hinge forms at the centre of the beam. Note that the beam material is assumed to have no strain hardening.

The second example considered, Example 5.2, is the clamped I beam shown in Fig. 5.7. The beam has the same material properties as those of Example 5.1.

The dimensions and finite element discretisation of the beam are given in Fig. 5.7; the load–displacement relationship at the beam centre is also provided. It is seen that there is an initial loss of stiffness corresponding to the

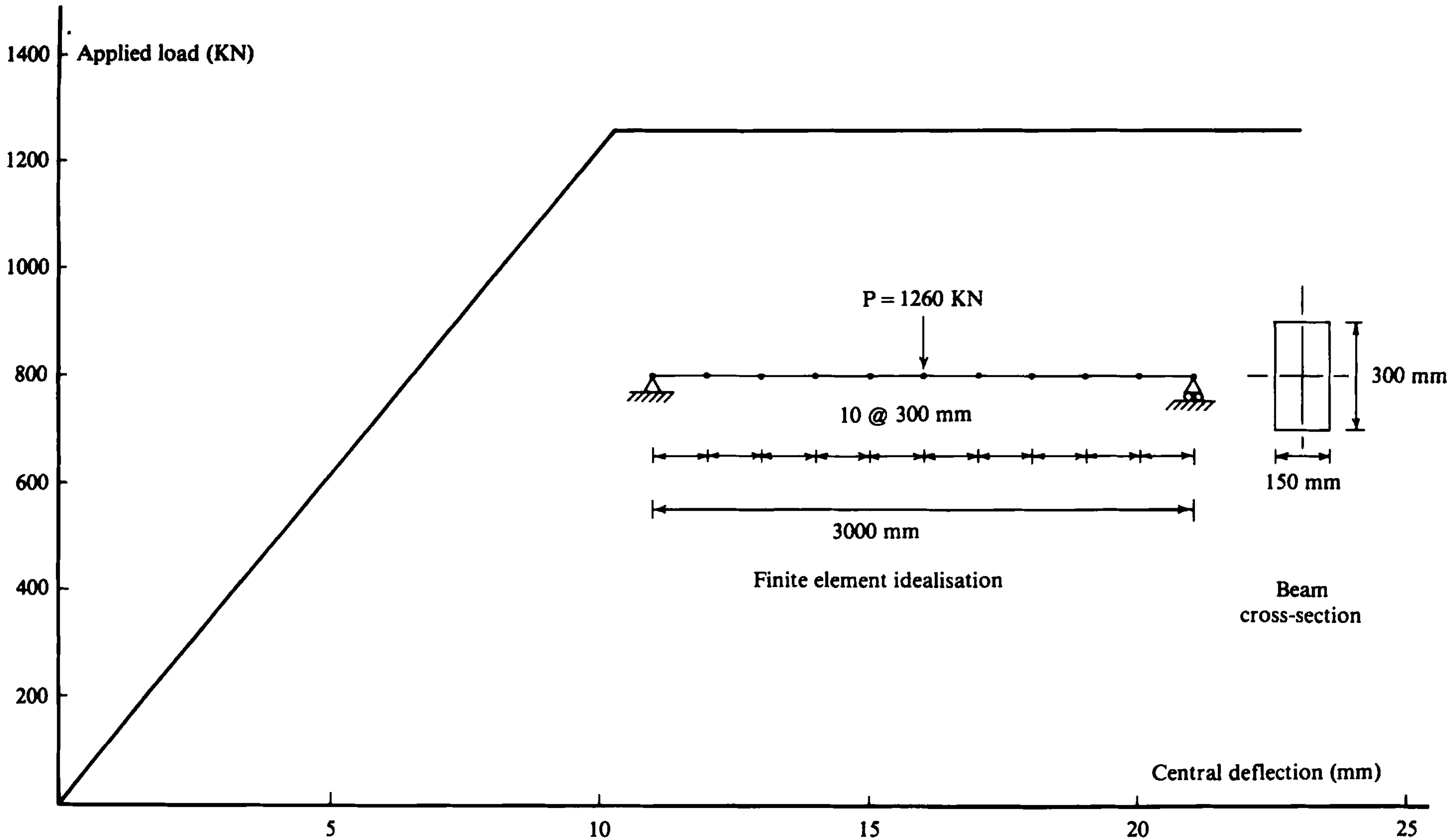


Fig. 5.6 Nonlayered elasto-plastic simply supported beam.

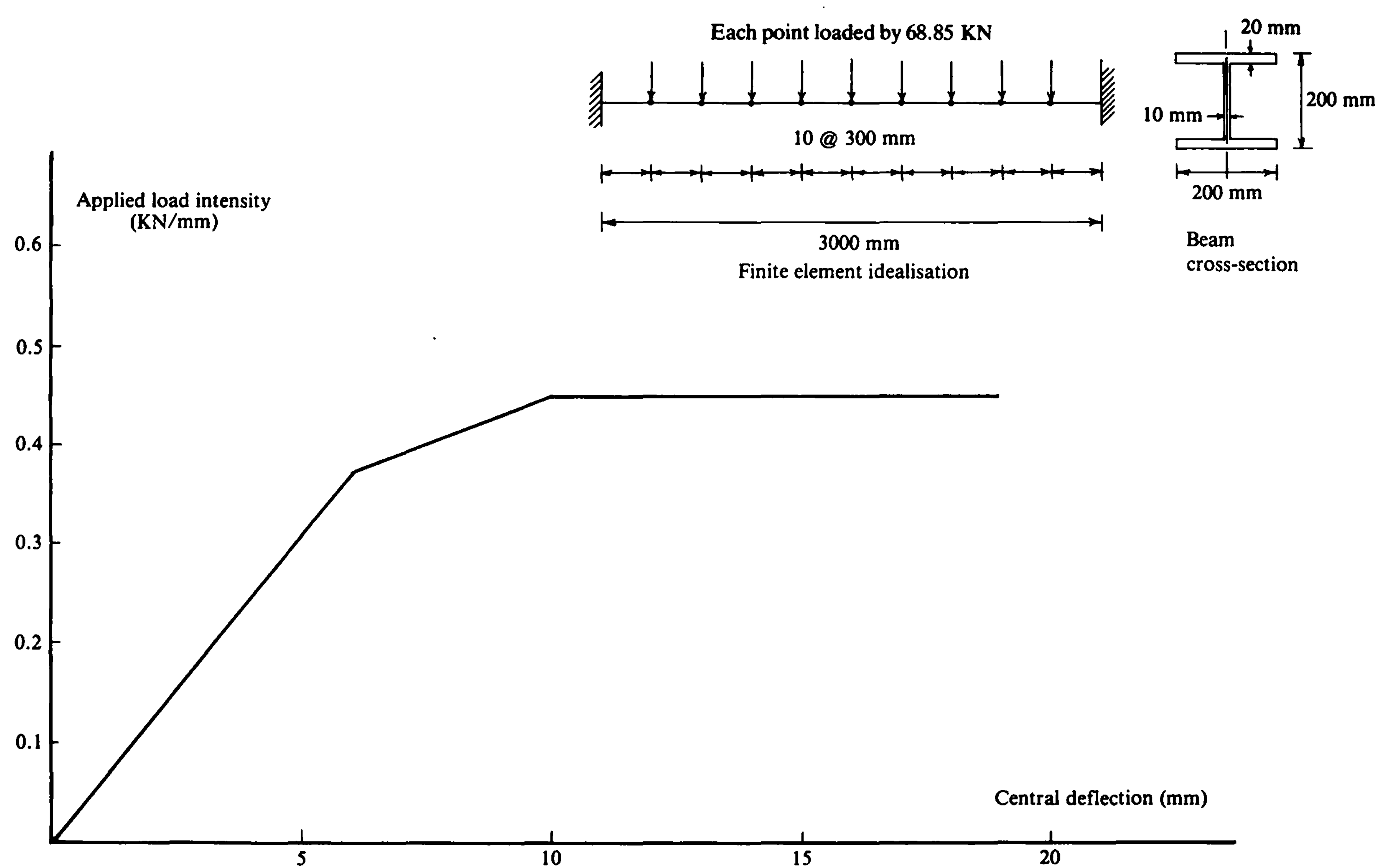


Fig. 5.7 Nonlayered elasto-plastic clamped beam.

yielding of the end sections followed by a further reduction when the central section becomes plastic resulting in a beam failure mechanism.

5.5 Elasto-plastic layered Timoshenko beams

5.5.1 Yielding of layered beams

In the 'layered' approach the beam or the plate is subdivided into a chosen number of layers, as shown in Fig. 5.8.

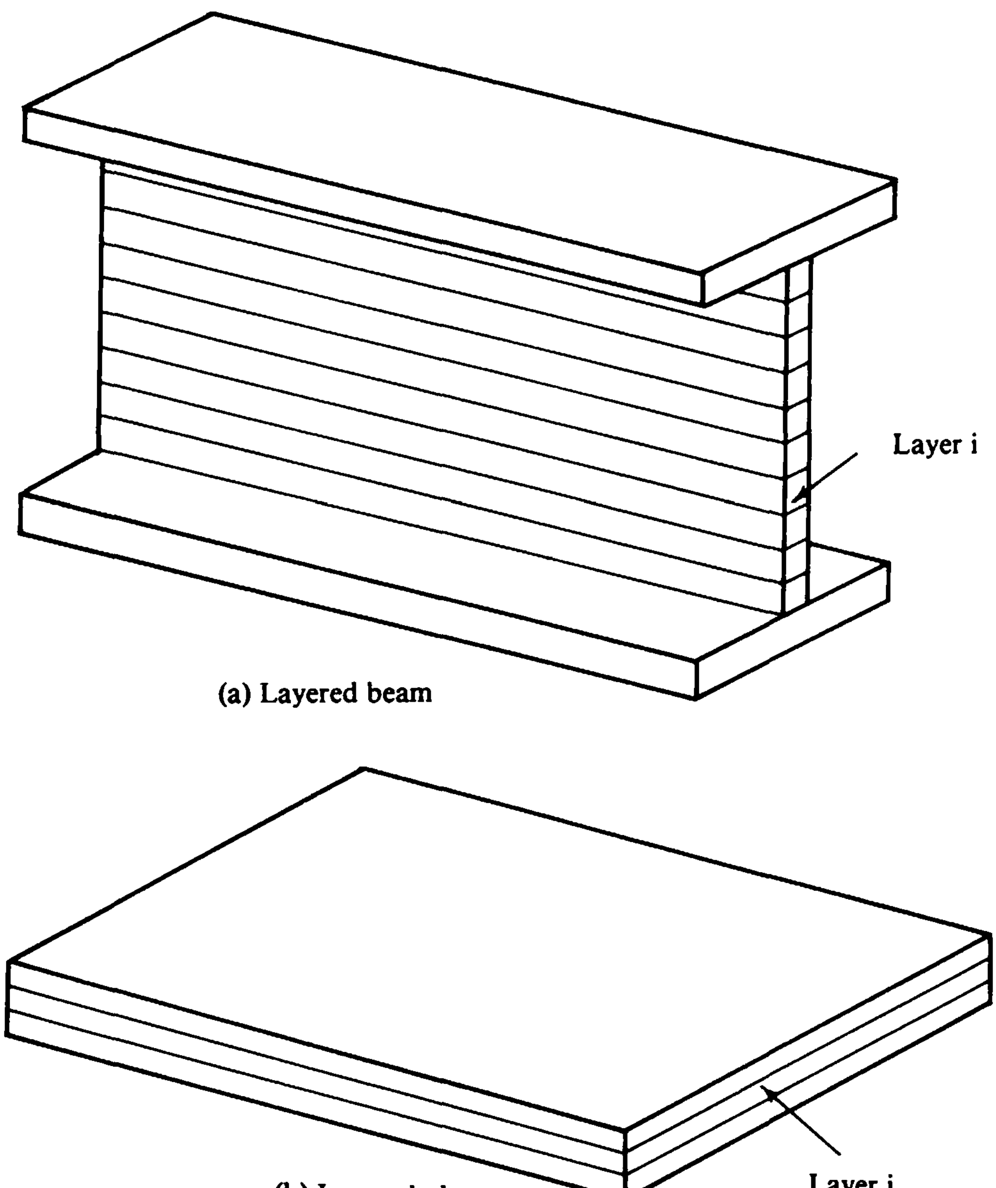


Fig. 5.8 Layered subdivision of beam and plate.

In the finite element solution it is assumed that as soon as the stress in the middle of the outer layers reaches the yield value, then the outer layers become plastic, while the rest of the layers remain elastic, as shown in

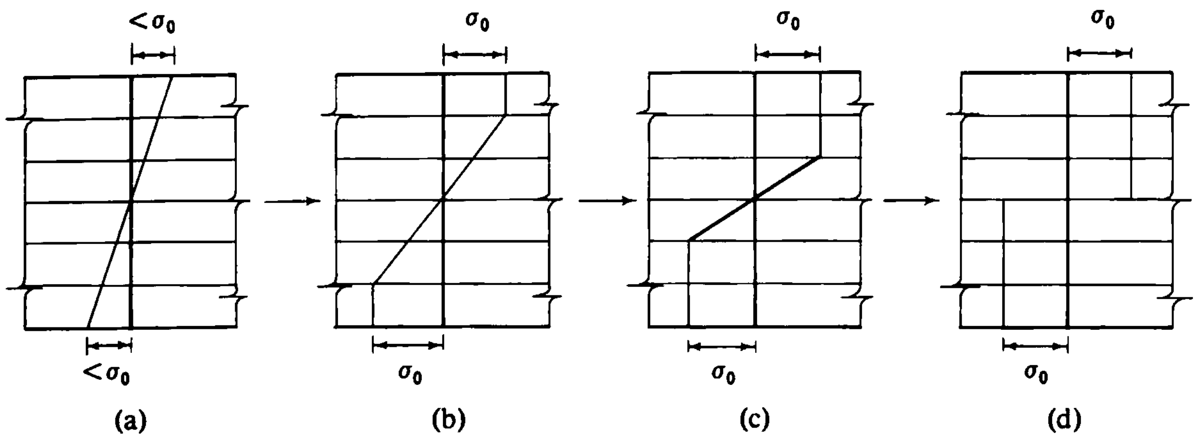


Fig. 5.9 Yielding of layered beam.

Fig. 5.9. Then, as plastification propagates, more layers become plastic, until the whole cross-section eventually becomes plastic.

5.5.2 Formation of the stiffness matrix in the layered approach

In the layered approach, we work in terms of stresses and not in terms of stress resultants as in the nonlayered approach. The state of stress at the middle of a layer is taken as representative for the entire layer.

Contributions to the stress resultants M and Q are found for each layer separately by integrating over the layer thickness only. The bending moments and shear forces are then found from the contributions of all the layers of the beam element.

The displacement field, stress–strain relationship and strain–displacement relationship are given in (5.1)–(5.10).

The virtual work expression is given by (5.11) and when we evaluate the bending moment M and shear force Q we use a mid-ordinate rule as follows:

$$M = EI\left(-\frac{d\theta}{dx}\right) \quad \text{and} \quad Q = G\hat{A}\,\epsilon_s \tag{5.48}$$

where

$$EI = \sum_l E_l b_l z_l^2 t_l \tag{5.49}$$

and

$$G\hat{A} = \sum_l G_l b_l t_l \tag{5.50}$$

and where b_l is the layer breadth
t_l is the layer thickness
z_l is the z-coordinate at the middle of the layer
E_l is the Young's modulus of the layer material
and G_l is the Shear modulus of the layer material.

However, if the stress at the middle surface of a layer reaches the uniaxial yield stress of the layer material, the whole layer is considered to be plastic and E_l is replaced by

$$E_l\left(1-\frac{E_l}{E_l+H'}\right),$$

where H' is the uniaxial strain hardening parameter. As mentioned before, the shear force–shear strain relationship is always elastic.

5.5.3 Solution of nonlinear equations

Recall that the virtual work expression (5.11) has the form

$$\int_0^l \int_{-t/2}^{t/2} \int_{b(-t/2)}^{b(t/2)} \left\{-z\frac{d(\delta\theta)}{dx}\sigma_x+\delta\beta\,\tau_{xz}\right\}dy\,dz\,dx-\int_0^l \delta w\,q\,dx = 0. \quad (5.51)$$

The mid-ordinate rule is again used to evaluate the first two integrals in (5.51) so that we obtain

$$[\delta\varphi]^T[p_f+p_s]-[\delta\varphi]^T f = 0 \quad (5.52)$$

where

$$p_f = \int_0^l [B_f]^T \bar{M}\,dx$$

and

$$p_s = \int_0^l [B_s]^T \bar{Q}\,dx$$

in which B_f, B_s and $\delta\varphi$ have been defined in (5.40), (5.41) and (5.43) respectively and in which

$$\bar{M} = \sum_l b_l\,\sigma_{xl}\,z_l\,t_l \quad (5.53)$$

and

$$\bar{Q} = \sum_l b_l\,\tau_{xzl}\,t_l. \quad (5.54)$$

Note that σ_{xl} and τ_{xzl} are the direct and shear stresses in the layer respectively. Since (5.52) is true for any arbitrary set of virtual displacements then

$$p_f+p_s-f = 0. \quad (5.55)$$

Contributions to p_f and p_s may be evaluated separately from each element so that

$$p_f^{(e)} = \int_{x_1^{(e)}}^{x_2^{(e)}} [B_f^{(e)}]^T \bar{M}^{(e)}\,dx = \int_{x_1^{(e)}}^{x_2^{(e)}} \left[0,\left(\frac{\bar{M}}{l}\right)^{(e)},0,-\left(\frac{\bar{M}}{l}\right)^{(e)}\right]^T dx$$
$$= [0,\bar{M}^{(e)},0,-\bar{M}^{(e)}]^T \quad (5.56)$$

and

$$
\begin{aligned}
p_s{}^{(e)} &= \int_{x_1^{(e)}}^{x_2^{(e)}} [B_s{}^{(e)}]^T \bar{Q}^{(e)}\, dx = \int_{x_1^{(e)}}^{x_2^{(e)}} \left[-\frac{1}{l^{(e)}},\ -\frac{1}{2},\ \frac{1}{l^{(e)}},\ -\frac{1}{2} \right]^T \bar{Q}^{(e)}\, dx \\
&= \left[-\bar{Q}^{(e)},\ -\frac{(\bar{Q}l)^{(e)}}{2},\ \bar{Q}^{(e)},\ -\frac{(\bar{Q}l)^{(e)}}{2} \right]^T .
\end{aligned}
\tag{5.57}
$$

The complete sequence of nonlinear equation solving is very similar to the one adopted in Table 5.1 for the nonlayered beam. Step 5 is now written as:

5. For each element evaluate for each layer $\sigma_{xl}{}^{(e)}$ and $\tau_{xzl}{}^{(e)}$. Check $\sigma_{xl}{}^{(e)}$ and adjust its value accordingly to account for any plastic behaviour. Evaluate the stress resultants $\bar{M}^{(e)}$ and $\bar{Q}^{(e)}$ and hence evaluate the residual force vector $[\psi^{(e)}]^{i+1} = p^{(e)} - f^{(e)}$. Assemble $[\psi^{(e)}]^{i+1}$ into the global residual force vector ψ^{i+1}.

5.5.4 Overall structure of layered beam program TIMLAY

The overall structure of the layered beam program is exactly the same as that of the nonlayered beam program given in Fig. 5.5. Subroutine STIFBL replaces STIFFB and subroutine RFORBL replaces REFORB. Subroutine STIFBL calls a further new routine called LAYER. The master routine BEML has minor changes as shown in the next section.

5.5.5 Modified and new routines

Master BEML This routine is almost identical to routine BEAM described earlier.

```
      MASTER BEML                                                       LYBM
C***********************************************************************LYBM
C                                                                       LYBM
C *** ELSTO-PLASTIC LAYERED TIMOSHENKO BEAM PROGRAM                     LYBM
C                                                                       LYBM
C***********************************************************************LYBM
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLAYR,NPROP,NNODE,IINCS,IITER,     LYBM
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,     LYBM
     .             NITER,NOUTP,FACTO                                    LYBM
      COMMON/UNIM2/PROPS(5,25),COORD(26),LNODS(25,2),IFPRE(52),         LYBM
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),       LYBM
     .             MATNO(25),STRES(25,2),PLAST(250),XDISP(52),          LYBM
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),      LYBM
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),          LYBM
     .             STRSL(250,2)                                         LYBM
      CALL  DATA                                                        LYBM
      CALL  INITAL                                                      LYBM
      DO 30 IINCS=1,NINCS                                               LYBM
      CALL  INCLOD                                                      LYBM
      DO 10 IITER=1,NITER                                               LYBM
      CALL  NONAL                                                       LYBM
      IF(KRESL.EQ.1) CALL  STIFBL                                       LYBM
      CALL  ASSEMB                                                      LYBM
      IF(KRESL.EQ.1) CALL GREDUC                                        LYBM
```

```
      IF(KRESL.EQ.2) CALL RESOLV
      CALL  BAKSUB
      CALL  RFORBL
      CALL  CONUND
      IF(NCHEK.EQ.0) GO TO 20
      IF(IITER.EQ.1.AND.NOUTP.EQ.1) CALL  RESULT
      IF(NOUTP.EQ.2) CALL  RESULT
   10 CONTINUE
      WRITE(6,900)
  900 FORMAT(1H0,5X,'SOLUTION NOT CONVERGED')
      STOP
   20 CALL  RESULT
   30 CONTINUE
      STOP
      END
```

Subroutine STIFBL This routine calculates the element stiffness matrices for the elasto-plastic layered Timoshenko beam element.

```
      SUBROUTINE  STIFBL
C**********************************************************************
C
C *** CALCULATES ELEMENT STIFFNESS MATRICES
C
C**********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLAYR,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO
      COMMON/UNIM2/PROPS(5,25),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(250),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),
     .             STRSL(250,2)
      REWIND 1
      DO 20 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      CALL  LAYER(IELEM,EIVAL,SVALU)
      HARDS=PROPS(LPROP,4)
      NODE1=LNODS(IELEM,1)
      NODE2=LNODS(IELEM,2)
      ELENG=ABS(COORD(NODE2)-COORD(NODE1))
      VALU1=0.5*SVALU
      VALU2=SVALU/ELENG
      VALU3=EIVAL/ELENG
      VALU4=0.25*SVALU*ELENG
      ESTIF(1,1)=VALU2
      ESTIF(1,2)=VALU1
      ESTIF(1,3)=-VALU2
      ESTIF(1,4)= VALU1
      ESTIF(2,2)= VALU3+VALU4
      ESTIF(2,3)= -VALU1
      ESTIF(2,4)= -VALU3+VALU4
      ESTIF(3,3)= VALU2
      ESTIF(3,4)=-VALU1
      ESTIF(4,4)= VALU3+VALU4
      DO 10 ISTIF=1,4
      DO 10 JSTIF=ISTIF,4
   10 ESTIF(JSTIF,ISTIF)=ESTIF(ISTIF,JSTIF)
      WRITE(1) ESTIF
   20 CONTINUE
      RETURN
      END
```

STBL 19 Call routine LAYER which evaluates approximate values of EI and exact values of $G\hat{A}$ using a mid-ordinate rule. Note that certain layers may be plastic.

Subroutine RFORBL This routine evaluates p for the layered beam using the mid-ordinate rule.

```
      SUBROUTINE RFORBL
C***********************************************************************
C
C *** CALCULATES INTERNAL EQUIVALENT NODAL FORCES
C
C***********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLAYR,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO
      COMMON/UNIM2/PROPS(5,25),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(250),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),
     .             STRSL(250,2)
      DIMENSION STRAN(2)
      DO 15 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=0.0
      DO 15 IDOFN=1,NDOFN
   15 STRES(IELEM,IDOFN)=0.0
      KLAYR=0
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      YOUNG=PROPS(LPROP,1)
      SHEAR=PROPS(LPROP,2)
      YIELD=PROPS(LPROP,3)
      HARDS=PROPS(LPROP,4)
      THKTO=PROPS(LPROP,5)
      NODE1=LNODS(IELEM,1)
      NODE2=LNODS(IELEM,2)
      ELENG=ABS(COORD(NODE2)-COORD(NODE1))
      WNOD1=XDISP(NODE1*NDOFN-1)
      WNOD2=XDISP(NODE2*NDOFN-1)
      THTA1=XDISP(NODE1*NDOFN)
      THTA2=XDISP(NODE2*NDOFN)
      STRAN(1)=(THTA1-THTA2)/ELENG
      STRAN(2)=(WNOD2-WNOD1)/ELENG
     .        -0.5*(THTA1+THTA2)
      ZMIDL=-THKTO/2.0
      KOUNT=5
      DO 50 ILAYR=1,NLAYR
      KLAYR=KLAYR+1
      KOUNT=KOUNT+1
      BRDTH=PROPS(LPROP,KOUNT)
      KOUNT=KOUNT+1
      THICK=PROPS(LPROP,KOUNT)
      ZMIDL=ZMIDL+THICK/2.0
      STLIN=YOUNG*STRAN(1)*ZMIDL
      STCUR=STRSL(KLAYR,1)+STLIN
      PREYS=YIELD+HARDS*ABS(PLAST(KLAYR))
      IF(ABS(STRSL(KLAYR,1)).GE.PREYS) GO TO 20
      ESCUR=ABS(STCUR)-PREYS
      IF(ESCUR.LE.0.0) GO TO 40
```

```
      RFACT=ESCUR/ABS(STLIN)
      GO TO 30
   20 IF(STRSL(KLAYR,1).GT.0.0.AND.STLIN.LE.0.0) GO TO 40
      IF(STRSL(KLAYR,1).LT.0.0.AND.STLIN.GE.0.0) GO TO 40
      RFACT=1.0
   30 REDUC=1.0-RFACT
      STRSL(KLAYR,1)=STRSL(KLAYR,1)+REDUC*STLIN+
     .         RFACT*YOUNG*(1.0-YOUNG/(YOUNG+HARDS))*STRAN(1)*ZMIDL
      PLAST(KLAYR)=PLAST(KLAYR)+RFACT*STRAN(1)*YOUNG/(YOUNG+HARDS)
     .*ZMIDL
      GO TO 45
   40 STRSL(KLAYR,1)=STRSL(KLAYR,1)+STLIN
   45 STRSL(KLAYR,2)=STRSL(KLAYR,2)+STRAN(2)*SHEAR
      STRES(IELEM,1)=STRES(IELEM,1)+STRSL(KLAYR,1)*
     .               BRDTH*THICK*ZMIDL
      STRES(IELEM,2)=STRES(IELEM,2)+STRSL(KLAYR,2)*
     .               BRDTH*THICK
      ZMIDL=ZMIDL+THICK/2.0
   50 CONTINUE
      ELOAD(IELEM,1)=ELOAD(IELEM,1)-STRES(IELEM,2)
      ELOAD(IELEM,2)=ELOAD(IELEM,2)+STRES(IELEM,1)
     .              -0.5*ELENG*STRES(IELEM,2)
      ELOAD(IELEM,3)=ELOAD(IELEM,3)+STRES(IELEM,2)
      ELOAD(IELEM,4)=ELOAD(IELEM,4)-STRES(IELEM,1)
     .              -0.5*ELENG*STRES(IELEM,2)
   70 CONTINUE
      RETURN
      END
```

Subroutine LAYER This routine evaluates EI and $G\hat{A}$ using the mid-ordinate rule. Note that certain layers may be plastic and therefore have a modified E.

```
      SUBROUTINE  LAYER(IELEM,EIVAL,SVALU)
C**********************************************************************
C
C *** CALCULATES INTEGRATED VALUES FOR EI AND GA THROUGH DEPTH
C
C**********************************************************************
      COMMON/UNIM1/NPOIN,NELEM,NBOUN,NLAYR,NPROP,NNODE,IINCS,IITER,
     .             KRESL,NCHEK,TOLER,NALGO,NSVAB,NDOFN,NINCS,NEVAB,
     .             NITER,NOUTP,FACTO
      COMMON/UNIM2/PROPS(5,25),COORD(26),LNODS(25,2),IFPRE(52),
     .             FIXED(52),TLOAD(25,4),RLOAD(25,4),ELOAD(25,4),
     .             MATNO(25),STRES(25,2),PLAST(250),XDISP(52),
     .             TDISP(26,2),TREAC(26,2),ASTIF(52,52),ASLOD(52),
     .             REACT(52),FRESV(1352),PEFIX(52),ESTIF(4,4),
     .             STRSL(250,2)
      EIVAL=0.0
      SVALU=0.0
      LPROP=MATNO(IELEM)
      KLAYR=(IELEM-1)*NLAYR
      SHEAR=PROPS(LPROP,2)
      HARDS=PROPS(LPROP,4)
      THKTO=PROPS(LPROP,5)
      ZMIDL=-THKTO/2.0
      KOUNT=5
      DO 10 ILAYR=1,NLAYR
      KLAYR=KLAYR+1
      YOUNG=PROPS(LPROP,1)
      IF(PLAST(KLAYR).NE.0.0) YOUNG=YOUNG*(1.0-YOUNG/(YOUNG+HARDS))
```

```
      KOUNT=KOUNT+1
      BRDTH=PROPS(LPROP,KOUNT)
      KOUNT=KOUNT+1
      THICK=PROPS(LPROP,KOUNT)
      ZMIDL=ZMIDL+THICK/2.0
      EIVAL=EIVAL+YOUNG*BRDTH*THICK*ZMIDL*ZMIDL
      SVALU=SVALU+SHEAR*BRDTH*THICK
      ZMIDL=ZMIDL+THICK/2.0
   10 CONTINUE
      RETURN
      END
```

5.5.6 Examples of layered elasto-plastic Timoshenko beam analysis

The third example considered in this chapter is the elasto-plastic analysis of the simple beam of Example 5.1. The layered solution is adopted in this case. A typical input data listing is provided in Appendix IV.

The results for both nonlayered and layered solutions to this beam problem are reproduced in Fig. 5.10.

The last example to be considered here is the layered solution of the clamped *I*-beam given in Example 5.1.

Again, both nonlayered and layered solution results are illustrated in Fig. 5.11.

From Figs. 5.10 and 5.11 it is obvious that the layered solution is more realistic. Yielding takes place gradually through the layers, resulting in smoother curves representing the load–displacement relationship.

5.6 Problems

5.1 Derive the main expressions for the elasto-plastic analysis of Timoshenko beams using elements with

(i) Quadratic shape functions

$$N_1{}^{(e)} = \frac{(x^{(e)} - x_2{}^{(e)})(x^{(e)} - x_3{}^{(e)})}{(x_1{}^{(e)} - x_2{}^{(e)})(x_1{}^{(e)} - x_3{}^{(e)})}$$

$$N_2{}^{(e)} = \frac{(x^{(e)} - x_1{}^{(e)})(x^{(e)} - x_3{}^{(e)})}{(x_2{}^{(e)} - x_1{}^{(e)})(x_2{}^{(e)} - x_3{}^{(e)})}$$

$$N_3{}^{(e)} = \frac{(x^{(e)} - x_1{}^{(e)})(x^{(e)} - x_2{}^{(e)})}{(x_3{}^{(e)} - x_1{}^{(e)})(x_3{}^{(e)} - x_2{}^{(e)})} \qquad (5.58)$$

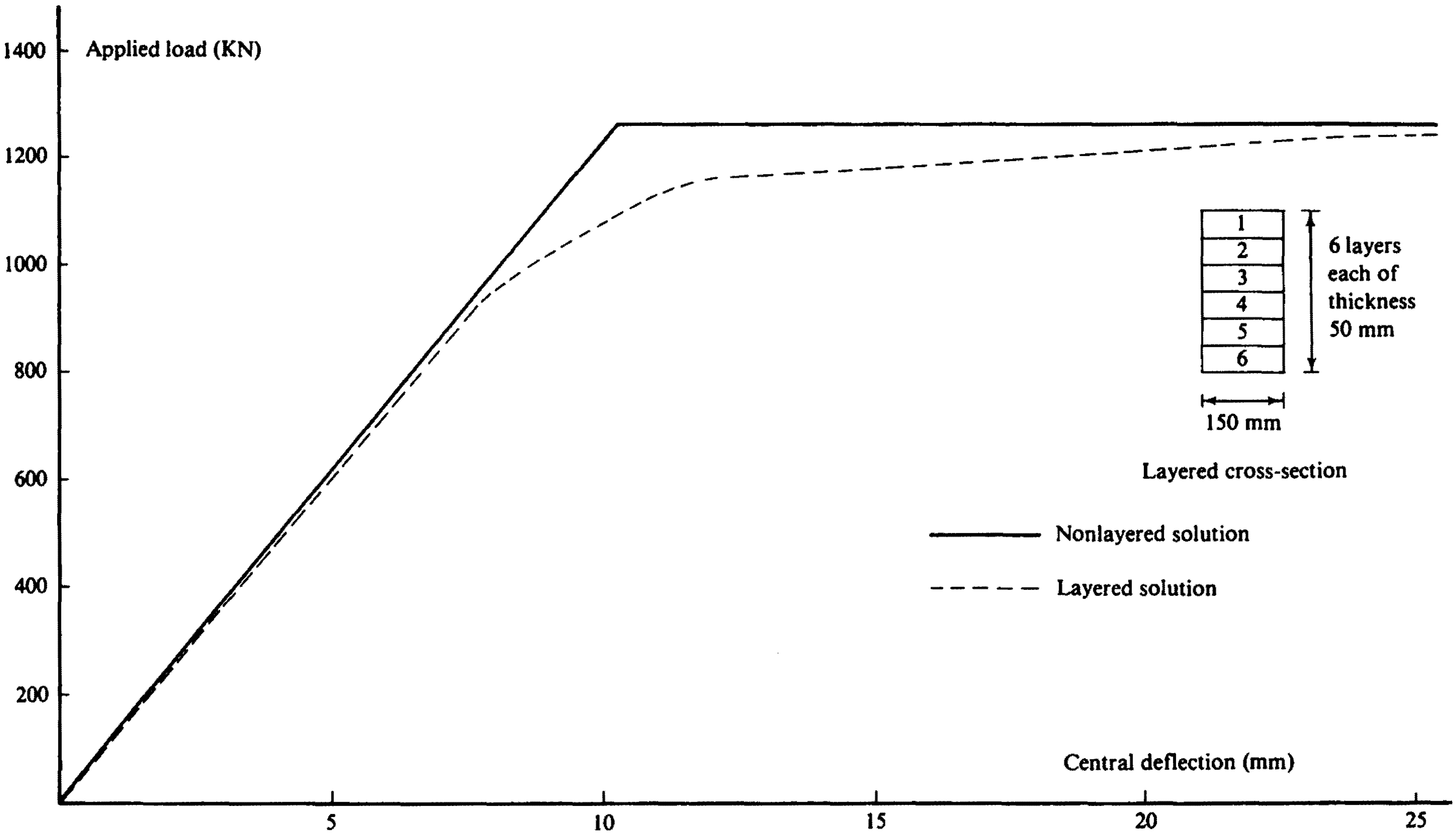


Fig. 5.10 Load-deflection diagrams for simply supported beam.

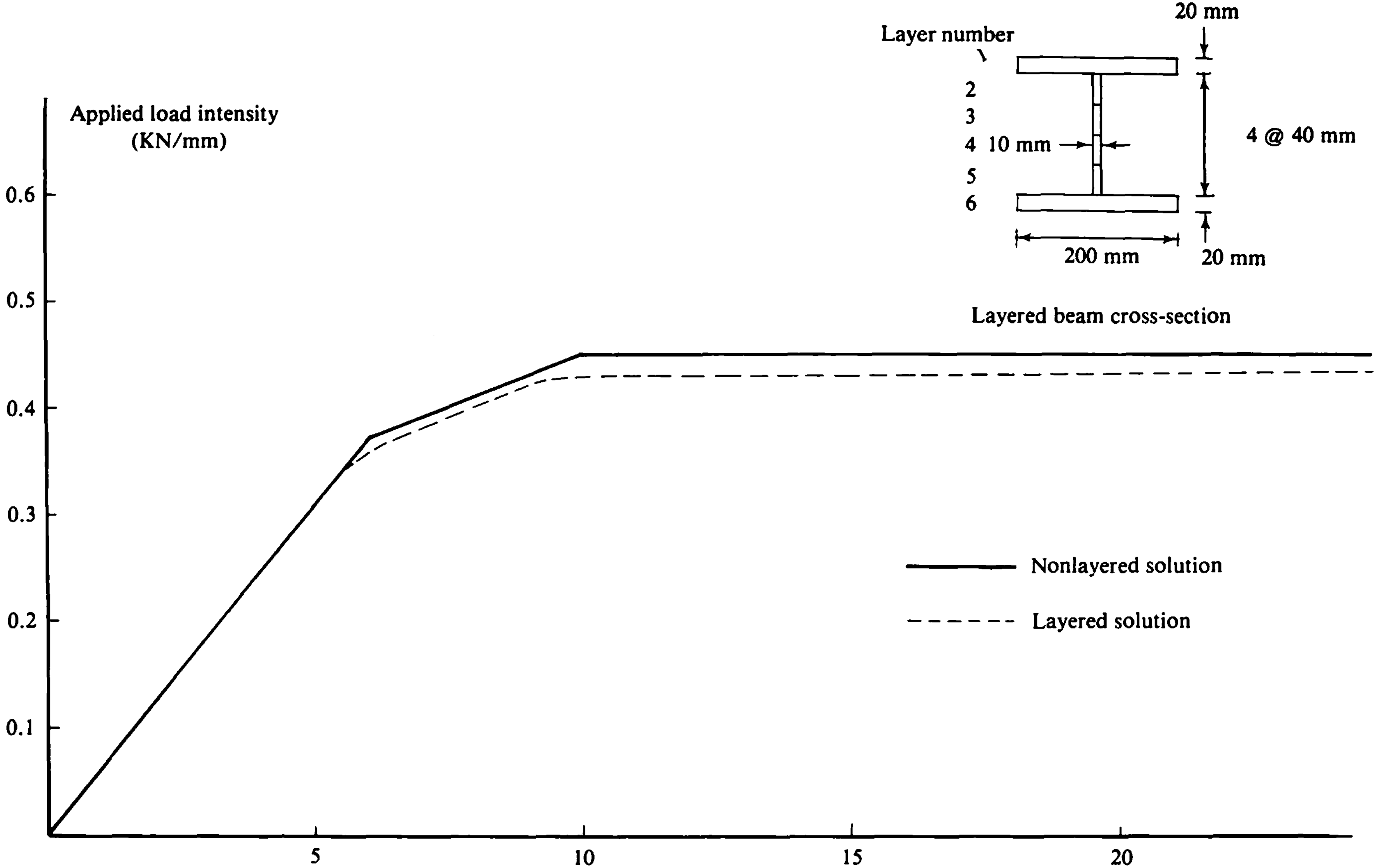


Fig. 5.11 Load-deflection diagrams for clamped beam.

(ii) Cubic shape functions

$$N_1^{(e)} = \frac{(x^{(e)}-x_2^{(e)})(x^{(e)}-x_3^{(e)})(x^{(e)}-x_4^{(e)})}{(x_1^{(e)}-x_2^{(e)})(x_1^{(e)}-x_3^{(e)})(x_1^{(e)}-x_4^{(e)})}$$

$$N_2^{(e)} = \frac{(x^{(e)}-x_1^{(e)})(x^{(e)}-x_3^{(e)})(x^{(e)}-x_4^{(e)})}{(x_2^{(e)}-x_1^{(e)})(x_2^{(e)}-x_3^{(e)})(x_2^{(e)}-x_4^{(e)})}$$

$$N_3^{(e)} = \frac{(x^{(e)}-x_1^{(e)})(x^{(e)}-x_2^{(e)})(x^{(e)}-x_4^{(e)})}{(x_3^{(e)}-x_1^{(e)})(x_3^{(e)}-x_2^{(e)})(x_3^{(e)}-x_4^{(e)})}$$

$$N_4^{(e)} = \frac{(x^{(e)}-x_1^{(e)})(x^{(e)}-x_2^{(e)})(x^{(e)}-x_3^{(e)})}{(x_4^{(e)}-x_1^{(e)})(x_4^{(e)}-x_2^{(e)})(x_4^{(e)}-x_3^{(e)})} \tag{5.59}$$

For the quadratic and cubic elements use 2-point and 3-point Gauss–Legendre integration rules respectively.

5.2 Develop a layered finite element Timoshenko beam program which allows for combined in-plane and bending behaviour of axially loaded beams or beams with cross-sections which are nonsymmetric about the neutral axis. Choose a displacement representation of the form

$$\bar{u}(x, z) = u_0(x) - z\theta_x(x) \tag{5.60}$$

in which $u_0(x)$ is the axial displacement at the neutral axis.

5.3 Use the concepts developed in Chapters 4 and 5 to develop the necessary relationships for layered and nonlayered elasto-viscoplastic Timoshenko beam analysis.

5.4 (i) Evaluate the additional stiffness terms required to represent the Winkler foundation by a 2-node linear Timoshenko beam element. For a foundation modulus k note that the additional virtual work term associated with the elastic foundation is

$$\int_0^l \delta w\, k w\, dx$$

in which δw is the virtual lateral displacement.

(ii) Modify programs TIMOSH and TIMLAY to allow for beams on elastic foundations.

(iii) Use the program to analyse a uniformly loaded, simply supported beam on a Winkler foundation. The elastic closed form solution for an Euler–Bernoulli beam predicts lateral displacements

$$w = \sum_{n=1,3,5,\ldots}^{\infty} \frac{4qL^4/(n^5\pi^5 EI)}{1+kL^4/(n^4\pi^4 EI)} \sin\frac{n\pi x}{L} \tag{5.61}$$

and bending moments

$$M = \sum_{n=1,3,5,\ldots}^{\infty} \frac{4qL^2/(n\pi)^3}{1+kL^4/(n^4\pi^4 EI)} \sin\frac{n\pi x}{L}. \tag{5.62}$$

Compare the elastic results from the modified programs with the above solution for various values of kL^4/EI and t/L where EI is the flexural rigidity, t is the thickness and L is the length of the beam.

(iv) For a given yield stress, σ_0, evaluate the ultimate load for various values of kL^4/EI and t/L.

5.5 (i) Consider the problem of finding the elastic deflections of a simply supported beam of length L, flexural rigidity EI, shear rigidity GA which is subjected to a uniform load q. The beam is elastically supported at mid-span by a single linear spring of stiffness K. Modify programs TIMOSH and TIMLAY to solve this problem. Check your finite element solutions by noting that the elastic Euler–Bernoulli solution is given as

$$w = \frac{4qL^4}{EI} \sum_{n=1,3,5,\ldots}^{\infty} \frac{\sin(n\pi x/L)}{n^5}$$

$$-\frac{2KSL^3}{\pi^4 EI} \sum_{n=1,3,5,\ldots}^{\infty} \left(\frac{\sin(n\pi/2)\sin(n\pi x/L)}{n^4}\right) \tag{5.63}$$

in which

$$S = \frac{5qL^4}{384EI} \bigg/ \left(1+\frac{KL^3}{48EI}\right). \tag{5.64}$$

(ii) When the load carried by the elastic support reaches a value F the supported beam becomes perfectly plastic. How can this be catered for in the modified version of TIMOSH and TIMLAY?

5.6 Use program TIMLAY to examine the effects of choosing
(i) different load incrementations
(ii) various convergence tolerances
(iii) various numbers of layers
on the example given in Section 5.4 and also Problems 5.4 and 5.5.

5.7 References

1. HINTON, E. and OWEN, D. R. J., *An Introduction to Finite Element Computations*, Pineridge Press, Swansea, U.K., 1979.

2. HUGHES, T. J. R., TAYLOR, R. L. and KANOKNUKULCHAI, S., A simple and efficient finite element for bending, *Int. J. Num. Meth. Engng.*, **11**, 1529–1543 (1977).
3. COWPER, G. R., The shear coefficient in Timoshenko's Beam Theory, *J. Appl. Mech.*, **33**, 335 (1966).
4. DYM, C. L. and SHAMES, I. H., *Solid Mechanics: A Variational Approach*, McGraw-Hill, New York, 1973.
5. HINTON, E. and OWEN, D. R. J., *Finite Element Programming*, Academic Press, London, 1977.

Part II

Chapter 6
Preliminary theory and standard subroutines for two-dimensional elasto-plastic applications

6.1 Introduction

In Part II of this text we extend the concepts and techniques developed in Part I for one-dimensional situations to now permit the solution of two-dimensional problems. In particular the following applications are presented:

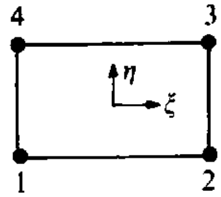
- Chapter 7 discusses the solution of elasto-plastic problems conforming to either plane stress, plane strain or axially symmetric conditions.
- Chapter 8 deals with plane stress/strain and axisymmetric problems where the material exhibits a time-dependent elasto-viscoplastic behaviour.
- Chapter 9 covers elasto-plastic plate bending situations.

The nonlinear algorithms developed in Chapter 2 will be employed in solution. These processes are general and the main modifications necessary are those appropriate to two-dimensional continuum theory or plate bending expressions which must now be used. For example the level of initial yielding will now be dependent on three or more independent stress components in place of the uniaxial case considered earlier.

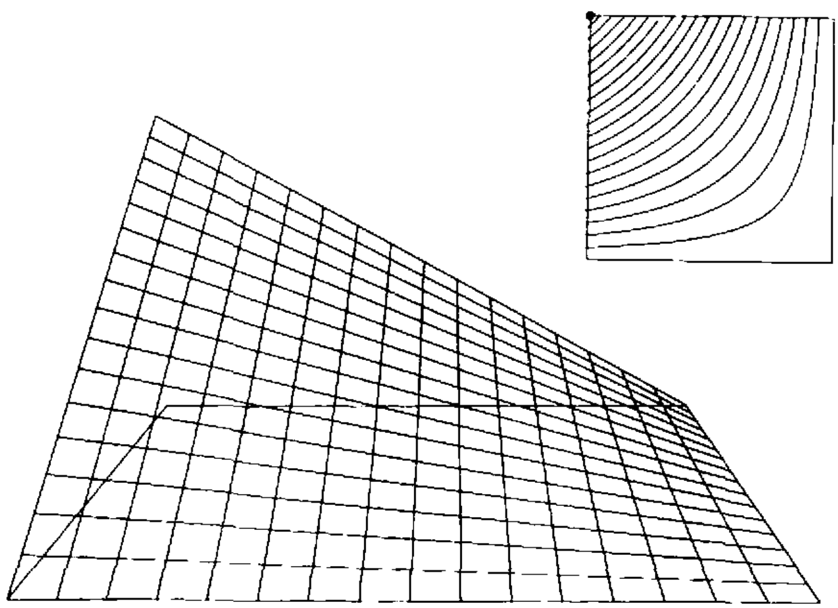
The development of an elasto-plastic stress analysis program requires all of the basic features of the corresponding elastic program. In particular the same basic element formulation is employed and a wide choice of element types is available. In this text we consider three different element types all based on an isoparametric formulation. The elements included are illustrated in Fig. 6.1 and are:

- The 4-node isoparametric quadrilateral element with linear displacement variation, Fig. 6.1(a).
- The 8-node Serendipity quadrilateral element with curved sides and a quadratic variation of the displacement field within the element, Fig. 6.1(b).
- The 9-node Lagrangian quadrilateral element which additionally has a central node, Fig. 6.1(c).

The basic theoretical expressions for these elements are provided in Section 6.3. The use of these higher order elements leads to particularly efficient



$$N_i(\xi, \eta) = \tfrac{1}{4}(1 + \xi\xi_i)(1 + \eta\eta_i).$$



Local node number	ξ_i	η_i
1	−1	−1
2	1	−1
3	1	1
4	−1	1

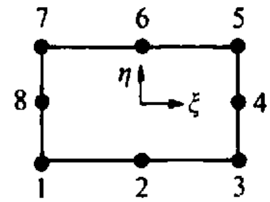
Fig. 6.1(a) The 4-node isoparametric quadrilateral element and shape functions.

elasto-plastic solution packages. In order to simplify matters as much as possible consideration is restricted to isotropic situations.*

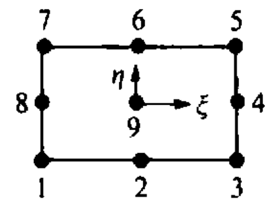
For all the plasticity applications presented in this text the classical incremental theory is employed with the full elasto-plastic material response being reproduced. Thus we are not concerned with limit state behaviour as predicted by rigid-plastic theories, etc.

Consideration is limited to small deformation situations where the strains can be assumed to be infinitesimal and Lagrangian and Eulerian geometric descriptions then coincide.

* Extension to orthotropic situations is feasible and has indeed been dealt with in Ref. 1.



8-node Serendipity element



9-node Lagrangian element

- for corner nodes

$$N_i^{(e)} = \tfrac{1}{4}(1+\xi\xi_i)(1+\eta\eta_i)(\xi\xi_i+\eta\eta_i-1), \quad i=1, 3, 5, 7,$$

- for midside nodes

$$N_i^{(e)} = \frac{\xi_i^2}{2}(1+\xi\xi_i)(1-\eta^2)+\frac{\eta_i^2}{2}(1+\eta\eta_i)(1-\xi^2), \quad i=2, 4, 6, 8.$$

Local node number	ξ_i	η_i
1	−1	−1
2	0	−1
3	1	−1
4	1	0
5	1	1
6	0	1
7	−1	1
8	−1	0
9	0	0

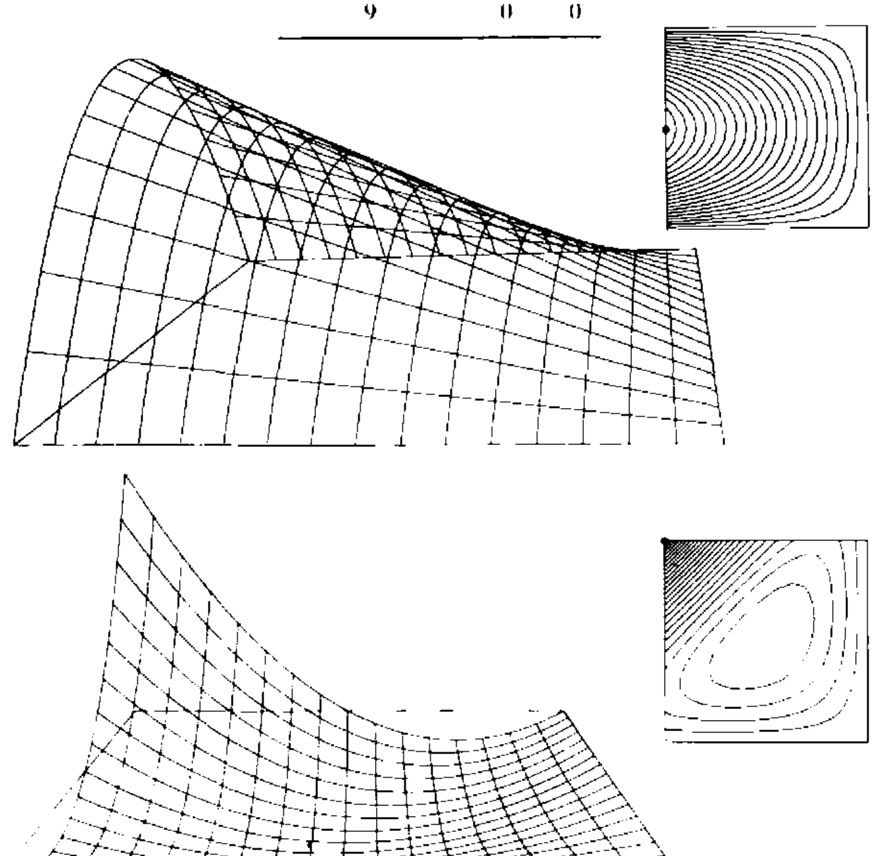
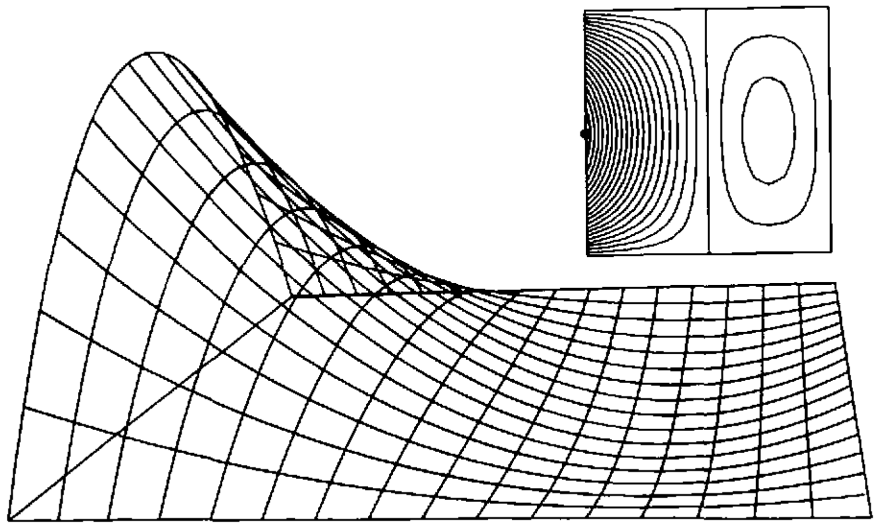


Fig. 6.1(b) The 8-node Serendipity quadrilateral element.



- for corner nodes

$$N_i^{(e)} = \tfrac{1}{4}(\xi^2 + \xi\xi_i)(\eta^2 + \eta\eta_i), \quad i = 1, 3, \quad 5, \quad 7,$$

- for midside nodes

$$N_i^{(e)} = \tfrac{1}{2}\eta_i^2(\eta^2 - \eta\eta_i)(1 - \xi^2) + \tfrac{1}{2}\xi_i^2(\xi^2 - \xi\xi_i)(1 - \eta^2), \quad i = 2, 4, 6, 8,$$

- for central node

$$N_i^{(e)} = (1 - \xi^2)(1 - \eta^2).$$

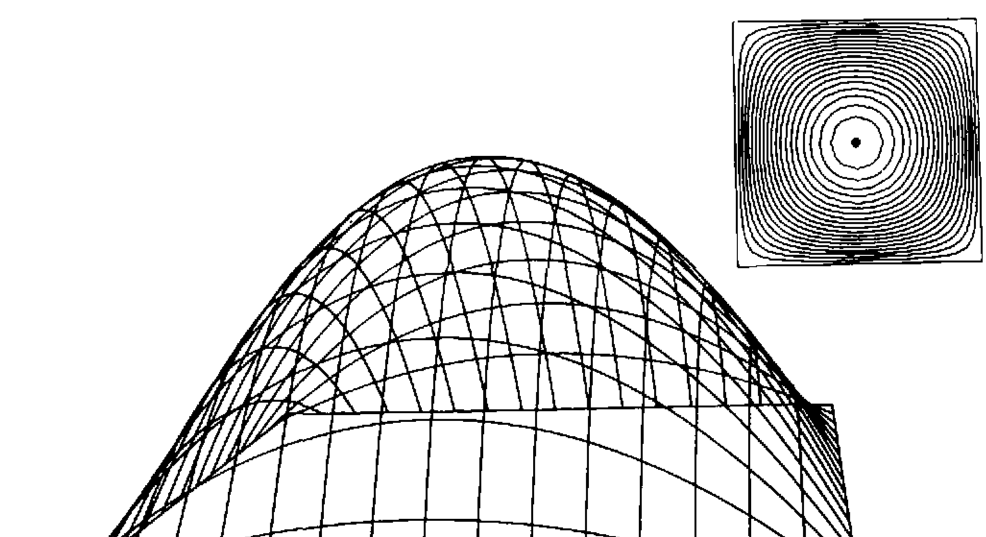


Fig. 6.1(c) The 9-node Lagrangian quadrilateral element.

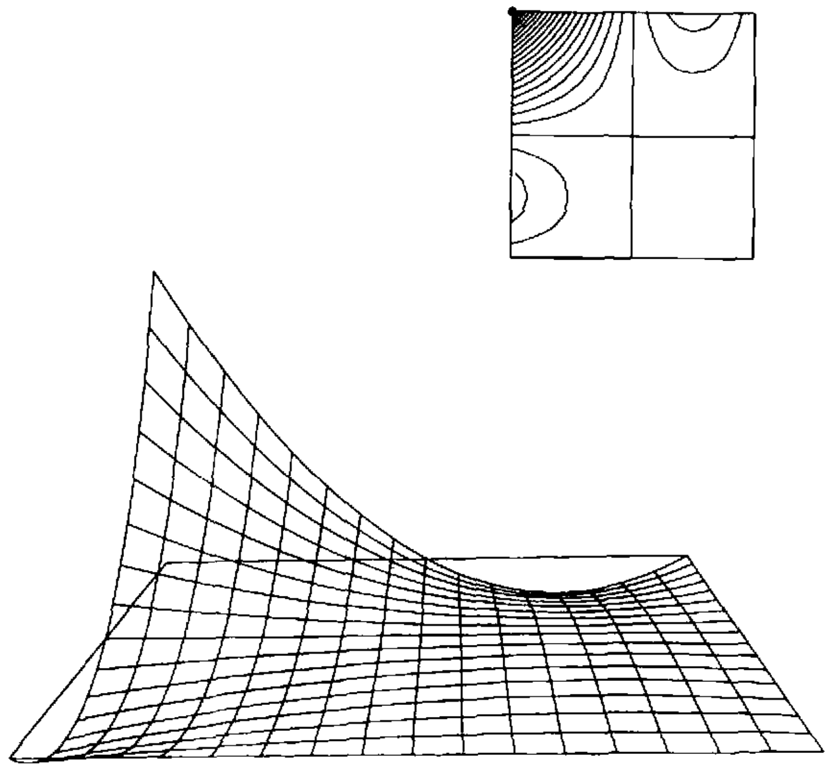


Fig. 6.1(c) The 9-node Lagrangian quadrilateral element (continued).

For each application, a computer code is developed which allows the solution of practical problems. The computation times of elasto-plastic problems are relatively high with solution costs being typically ten times those of the corresponding linear elastic analysis. Of course a direct comparison would depend on the extent of plastic yielding and how close to the ultimate load carrying capacity a solution is sought. In view of these relatively high computer costs it is essential that the codes developed should be as efficient as possible and that any numerical techniques which reduce the computational requirements be employed. Since the main aim of this text is to fulfil a teaching role some compromise must however be inevitably made between program clarity and efficiency. The applicability of the programs presented is demonstrated by the solution of practical examples. Detailed user instructions for all of the computer programs presented in Part II of this text are provided in Appendix II.

In Section 6.2 the basic expressions for the linear elastic finite element analysis of two-dimensional continua and plate bending problems are presented. Section 6.3 outlines the principles of isoparametric element formulation with particular attention being given to the role of numerical integration. Standard subroutines pertaining to linear elastic finite element analysis are reviewed in Section 6.4 and some subroutines common to the three nonlinear applications considered in Chapters 7, 8 and 9 are presented in Section 6.5.

6.2 Virtual work expressions for various solid mechanics applications

6.2.1 Introduction

In this section we briefly describe various two-dimensional solid mechanics finite element applications in the elastic range only. Later in Chapters 7–9 we demonstrate how elasto-plastic or elasto-viscoplastic behaviour may be included in these applications using finite elements.

In Part I we presented some very simple finite element representations. By contrast, in Part II we include numerically integrated isoparametric quadrilateral elements.

6.2.2 Virtual work expression

If a body is subjected to a set of body forces $\boldsymbol{b}$ then by the Virtual Work Principle we can write

$$\int_{\Omega} [\delta\boldsymbol{\epsilon}]^T \boldsymbol{\sigma}\, d\Omega - \int_{\Omega} [\delta\boldsymbol{u}]^T \boldsymbol{b}\, d\Omega - \int_{\Gamma_t} [\delta\boldsymbol{u}]^T \boldsymbol{t}\, d\Gamma = 0, \tag{6.1}$$

where $\boldsymbol{\sigma}$ is the vector of stresses, $\boldsymbol{t}$ is the vector of boundary tractions, $\delta\boldsymbol{u}$ is the vector of virtual displacements, $\delta\boldsymbol{\epsilon}$ is the vector of associated virtual strains, Ω is the domain of interest, Γ_t is that part of the boundary on which boundary tractions are prescribed and Γ_u is that part of the boundary on which displacements are prescribed.

6.2.3 Plane stress

Consider some typical plane stress problems shown in Fig. 6.2. Typically a thin plate is subjected to loads applied in the xy plane, that is the plane of the structure.[2] The thickness of the plate is assumed to be small compared with the plan dimensions in the xy plane. Stresses are assumed to be constant through the thickness of the plate and σ_z, τ_{zx} and τ_{zy} are ignored. Thus the displacements may now be expressed as

$$\boldsymbol{u} = [u, v]^T, \tag{6.2}$$

where u and v are the in-plane displacements in the x and y directions respectively.

The strain components may be listed in the vector

$$\boldsymbol{\epsilon} = [\epsilon_x, \epsilon_y, \gamma_{xy}]^T, \tag{6.3}$$

where for small displacements the normal strains are given as

$$\epsilon_x = \frac{\partial u}{\partial x}, \qquad \epsilon_y = \frac{\partial v}{\partial y},$$

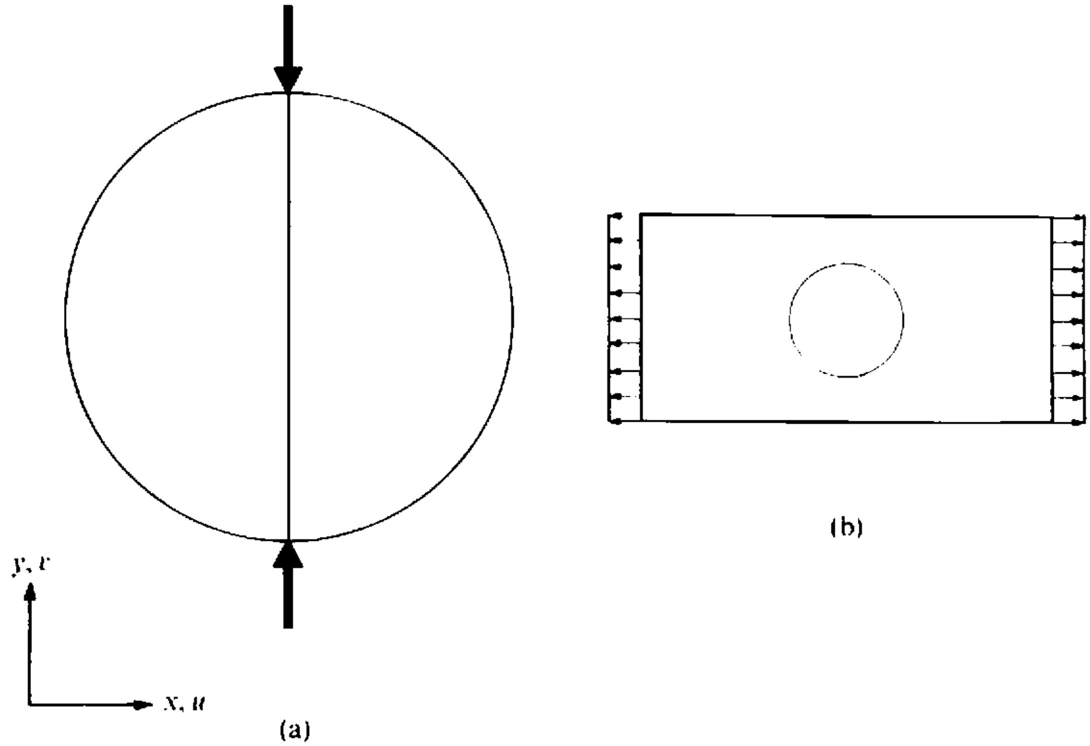


Fig. 6.2 Typical plane stress problems.

and the shear strain is given as

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}.$$

Note that virtual displacements are listed in the vector

$$\delta \boldsymbol{u} = [\delta u, \delta v]^T, \tag{6.4}$$

and the associated virtual strains are

$$\delta \boldsymbol{\epsilon} = \left[\frac{\partial(\delta u)}{\partial x}, \frac{\partial(\delta v)}{\partial y}, \frac{\partial(\delta u)}{\partial y} + \frac{\partial(\delta v)}{\partial x}\right]^T. \tag{6.5}$$

The relevant stress–strain relationships may be written as

$$\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\epsilon}, \tag{6.6}$$

where

$$\boldsymbol{\sigma} = [\sigma_x, \sigma_y, \tau_{xy}]^T,$$

in which σ_x and σ_y are the normal stresses and τ_{xy} is the shear stress.

For linear elastic situations the stress–strain or constitutive matrix is given as

$$\boldsymbol{D} = \frac{E}{(1-\nu^2)}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{(1-\nu)}{2} \end{bmatrix}, \tag{6.7}$$

in which E and ν are the elastic modulus and Poisson's ratio respectively.

The body forces $\boldsymbol{b}$ are written as

$$\boldsymbol{b} = [b_x, b_y]^T, \tag{6.8}$$

in which b_x and b_y are the body forces per unit volume in the x and y directions respectively.

Boundary tractions $\boldsymbol{t}$ may be expressed as

$$\boldsymbol{t} = [t_x, t_y]^T, \tag{6.9}$$

in which t_x and t_y are the boundary tractions per unit length.

An element of volume $d\Omega$ is given as

$$d\Omega = t\,dx\,dy, \tag{6.10}$$

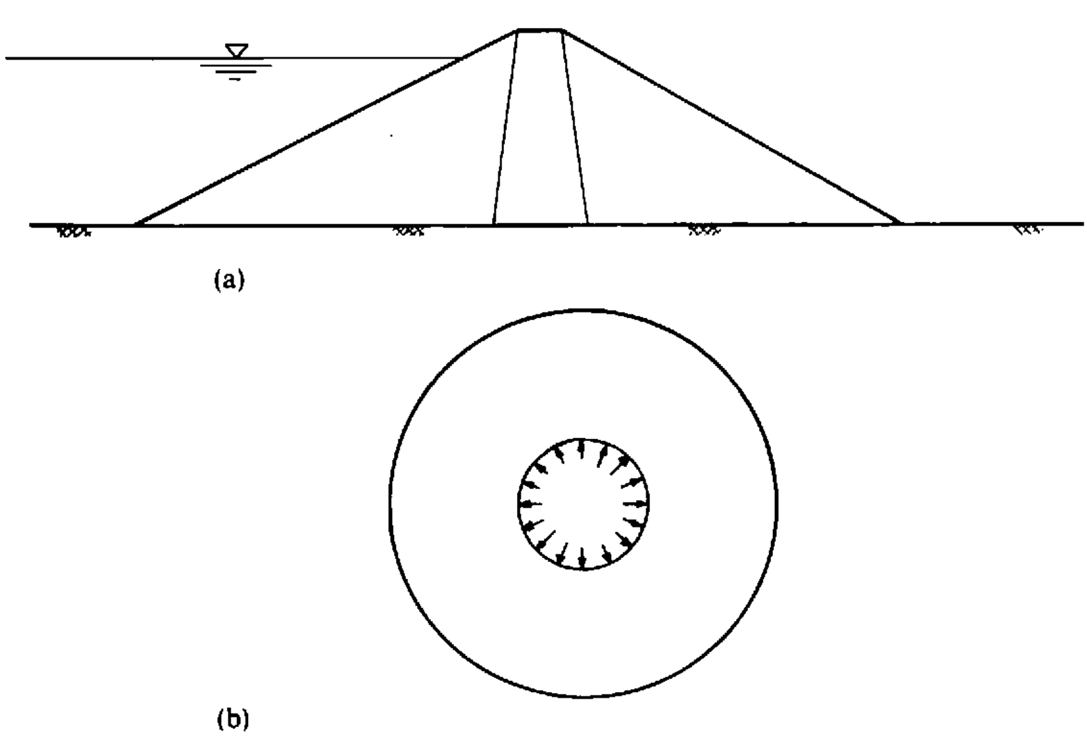
where t is the plate thickness.

6.2.4 Plane strain

For plane strain problems the thickness dimension normal to a certain plane (say the xy plane) is large compared with the typical dimensions in the xy plane and the body is subjected to loads in the xy plane only. For plane strain problems[2] it may be assumed that the displacements in the z direction are negligible and that the in-plane displacements u and v are independent of z. Figure 6.3 illustrates some typical plane strain problems.

The displacements are then listed in the vector

$$\boldsymbol{u} = [u, v]^T, \tag{6.11}$$



(a)

(b)

Fig. 6.3 Typical plane strain problems.

in which u and v are the in-plane displacements in the x and y directions respectively.

The in-plane strain components may be expressed as

$$\boldsymbol{\epsilon} = [\epsilon_x, \epsilon_y, \gamma_{xy}]^T, \tag{6.12}$$

where ϵ_x, ϵ_y and γ_{xy} have the same meaning as the strain components in plane stress applications.

Again the virtual displacements and associated virtual strains are respectively given as

$$\delta\boldsymbol{u} = [\delta u, \delta v]^T, \tag{6.13}$$

and

$$\delta\boldsymbol{\epsilon} = \left[\frac{\partial(\delta u)}{\partial x}, \frac{\partial(\delta v)}{\partial y}, \frac{\partial(\delta u)}{\partial y}+\frac{\partial(\delta v)}{\partial x}\right]^T. \tag{6.14}$$

The stress–strain relationships may be written in the form

$$\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\epsilon}, \tag{6.15}$$

where the stresses $\boldsymbol{\sigma} = [\sigma_x, \sigma_y, \tau_{xy}]^T$ have the same meaning as the stresses in plane stress applications.

For linear elastic materials the stress–strain or constitutive matrix $\boldsymbol{D}$ is given as

$$\boldsymbol{D} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} (1-\nu) & \nu & 0 \\ \nu & (1-\nu) & 0 \\ 0 & 0 & \dfrac{(1-2\nu)}{2} \end{bmatrix}. \tag{6.16}$$

Note that the stress normal to the xy plane is nonzero and may be evaluated as

$$\sigma_z = \nu(\sigma_x + \sigma_y). \tag{6.17}$$

The body forces $\boldsymbol{b}$ and surface tractions $\boldsymbol{t}$ have the same meaning as those adopted for plane stress problems.

A typical element of volume is given as

$$d\Omega = dx\, dy. \tag{6.18}$$

under the assumption that a unit slice of the problem is being analysed.

6.2.5 Axisymmetric solids

For a three-dimensional solid which is symmetrical about its centreline axis (which coincides with the z axis) and which is subjected to loads and boundary conditions that are symmetrical about this axis, then the behaviour[2] is independent of the circumferential coordinate θ. Figure 6.4 shows a typical axisymmetric solid.

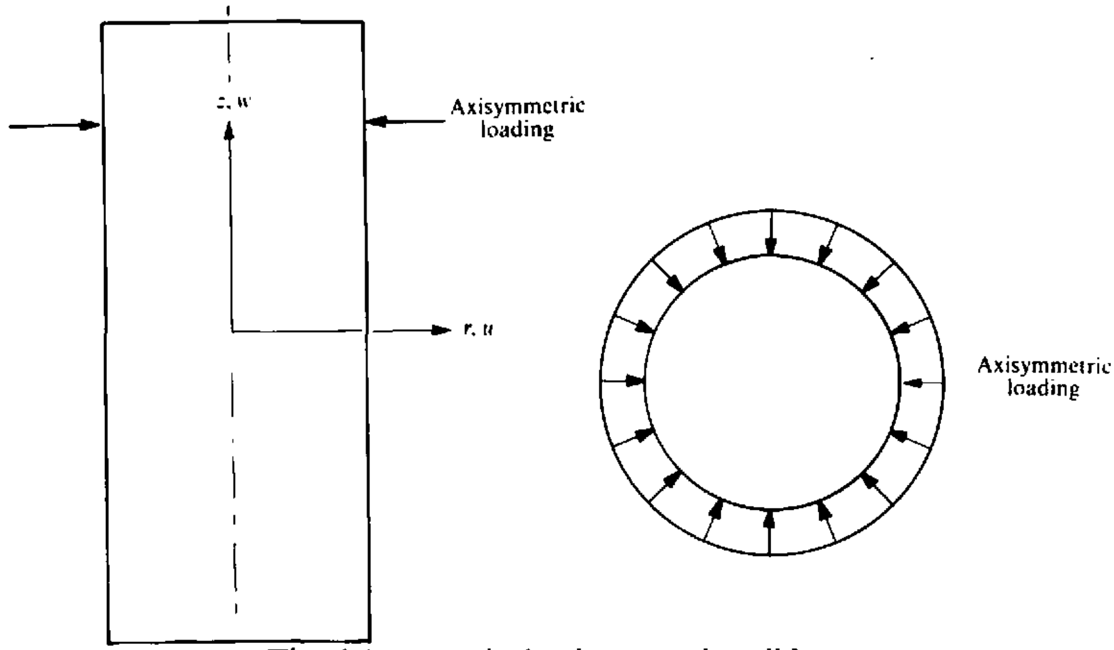


Fig. 6.4 A typical axisymmetric solid.

The displacements may here be expressed as

$$\boldsymbol{u} = [u, w]^T, \tag{6.19}$$

where u and w are the displacements in the r and z directions respectively.

The nonzero strains are given as

$$\boldsymbol{\epsilon} = [\epsilon_r, \epsilon_\theta, \epsilon_z, \gamma_{rz}]^T, \tag{6.20}$$

where for small displacements, the normal strains are given as

$$\epsilon_r = \frac{\partial u}{\partial r}, \quad \epsilon_\theta = \frac{u}{r} \quad \text{and} \quad \epsilon_z = \frac{\partial w}{\partial z},$$

and the shear strain is

$$\gamma_{rz} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial r}.$$

Virtual displacements and associated virtual strains are respectively given as

$$\delta\boldsymbol{u} = [\delta u, \delta w]^T, \tag{6.21}$$

and

$$\delta\boldsymbol{\epsilon} = \left[\frac{\partial(\delta u)}{\partial r}, \frac{\delta u}{r}, \frac{\partial(\delta w)}{\partial z}, \frac{\partial(\delta u)}{\partial z} + \frac{\partial(\delta w)}{\partial r}\right]^T. \tag{6.22}$$

The stress–strain relationships are given as

$$\boldsymbol{\sigma} = \boldsymbol{D}\,\boldsymbol{\epsilon}, \tag{6.23}$$

where $\boldsymbol{\sigma} = [\sigma_r, \sigma_\theta, \sigma_z, \tau_{rz}]^T$, in which σ_r, σ_θ and σ_z are the normal stresses in the r, θ and z directions respectively and τ_{rz} is the shear stress in the rz plane.

For linear elastic materials, the stress–strain matrix is given as

$$D = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} (1-\nu) & \nu & 0 & 0 \\ \nu & (1-\nu) & \nu & 0 \\ 0 & \nu & (1-\nu) & 0 \\ 0 & 0 & 0 & \dfrac{(1-2\nu)}{2} \end{bmatrix} \tag{6.24}$$

The body forces are given as

$$\boldsymbol{b} = [b_r, b_z]^T, \tag{6.25}$$

where b_r and b_z are the body forces/unit volume in the r and z direction respectively.

The boundary tractions may be expressed as

$$\boldsymbol{t} = [t_r, t_z]^T, \tag{6.26}$$

where t_r and t_z are the boundary tractions/unit surface in the r and z directions.

An elemental volume is given as

$$d\Omega = 2\pi r\, dr\, dz. \tag{6.27}$$

6.2.6 Mindlin plates

In Mindlin plate theory it is possible to allow for transverse shear deformation. It thus offers an alternative to classical Kirchhoff thin plate theory. The main assumptions are that:

(a) displacements are small compared with the plate thickness,
(b) the stress normal to the midsurface of the plate is negligible,
(c) normals to the midsurface before deformation remain straight but not necessarily normal to the midsurface after deformation.

A typical Mindlin plate is shown in Fig. 6.5. Note that Mindlin plate theory is the two-dimensional equivalent of Timoshenko beam theory which was discussed in Chapter 5.

The main displacement parameters can be expressed

$$\boldsymbol{u} = [w, \theta_x, \theta_y]^T, \tag{6.28}$$

in which w is the lateral plate displacement normal to the xy plane and variables θ_x and θ_y are the normal rotations in the xz and yz planes. Here it should be noted that

$$\theta_x = \frac{\partial w}{\partial x} - \phi_x \quad \text{and} \quad \theta_y = \frac{\partial w}{\partial y} - \phi_y, \tag{6.29}$$

where θ_x and θ_y are the rotations of the normal in the xz and yz planes

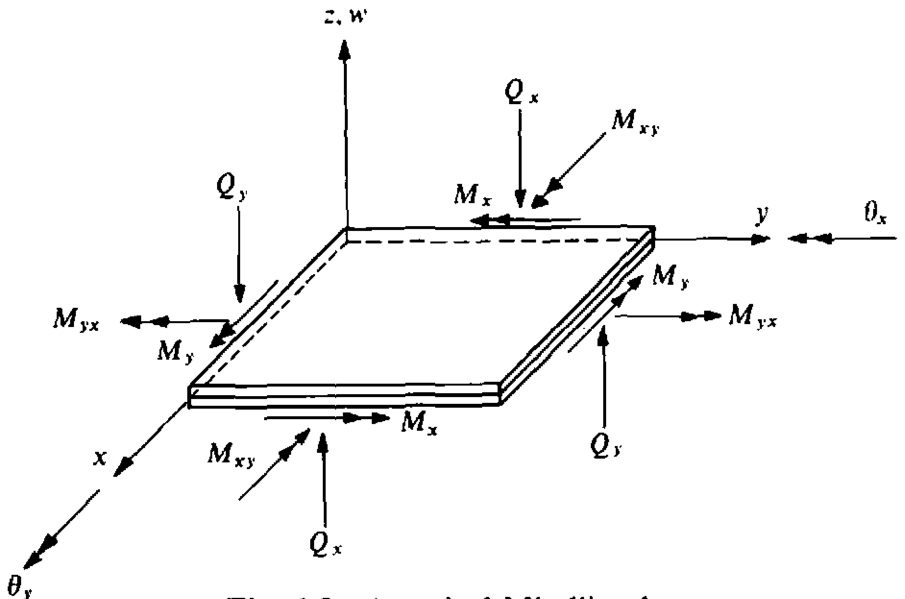


Fig. 6.5 A typical Mindlin plate.

respectively and are integrated measures of the transverse shear strain. In thin plate theory it is assumed that shear rotations ϕ_x and ϕ_y, defined below, are equal to zero.

The strains, or more exactly the strain resultants, may be expressed as

$$\epsilon = [r_x, r_y, r_{xy}, \phi_x, \phi_y]^T, \tag{6.30}$$

where the curvatures are given as

$$r_x = -\frac{\partial \theta_x}{\partial x} \quad \text{and} \quad r_y = -\frac{\partial \theta_y}{\partial y},$$

and the twisting curvature is

$$r_{xy} = -\left(\frac{\partial \theta_y}{\partial x}+\frac{\partial \theta_x}{\partial y}\right).$$

The shear strains are expressed as

$$\phi_x = \left(\frac{\partial w}{\partial x}-\theta_x\right) \quad \text{and} \quad \phi_y = \left(\frac{\partial w}{\partial y}-\theta_y\right). \tag{6.31}$$

Virtual displacements and rotations and associated virtual curvatures and shear strains are respectively given as

$$\delta \boldsymbol{u} = [\delta w, \delta\theta_x, \delta\theta_y]^T, \tag{6.32}$$

and

$$\delta\epsilon = \left[-\frac{\partial(\delta\theta_x)}{\partial x}, \; -\frac{\partial(\delta\theta_y)}{\partial y}, \; -\frac{\partial(\delta\theta_x)}{\partial y}-\frac{\partial(\delta\theta_y)}{\partial x}, \right.$$
$$\left. \frac{\partial(\delta w)}{\partial x}-\delta\theta_x, \; \frac{\partial(\delta w)}{\partial y}-\delta\theta_y\right]^T. \tag{6.33}$$

The constitutive relationships are given in the form

$$\sigma = D\,\epsilon, \tag{6.34}$$

where

$$\sigma = [M_x, M_y, M_{xy}, Q_x, Q_y]^T,$$

in which M_x and M_y are the direct bending moments and M_{xy} is the twisting moment. The quantities Q_x and Q_y are the shear forces in the xz and yz planes.

For an isotropic elastic material

$$D = \begin{bmatrix} D & \nu D & 0 & 0 & 0 \\ \nu D & D & 0 & 0 & 0 \\ 0 & 0 & \dfrac{(1-\nu)}{2}D & 0 & 0 \\ 0 & 0 & 0 & S & 0 \\ 0 & 0 & 0 & 0 & S \end{bmatrix}, \tag{6.35}$$

in which for a plate of thickness t

$$D = \frac{Et^3}{12(1-\nu^2)} \quad \text{and} \quad S = \frac{Gt}{1.2},$$

where G is the shear modulus and the factor 1.2 is a shear correction term.

Here we will not consider surface tractions. For a more complete discussion of this and other aspects of Mindlin plate theory the reader is directed to the work of Hughes and his coworkers.[3] We will only consider body forces of the form

$$b = [q, 0, 0]^T, \tag{6.36}$$

where q is the lateral distributed loading per unit area.

An elemental plate area is given as

$$d\Omega = dx\,dy. \tag{6.37}$$

6.3 Isoparametric finite element representation

6.3.1 Governing equations

In this section we present the discretised governing equations for the solid mechanics applications described in Sections 6.2.3–6.2.6. In a finite element representation, the displacements and strains and their virtual counterparts may be expressed by the relationships

$$u = \sum_{i=1}^{n} N_i\, d_i, \qquad \delta u = \sum_{i=1}^{n} N_i\, \delta d_i, \tag{6.38}$$

$$\boldsymbol{\epsilon} = \sum_{i=1}^{n} \boldsymbol{B}_i \boldsymbol{d}_i, \qquad \delta\boldsymbol{\epsilon} = \sum_{i=1}^{n} \boldsymbol{B}_i \,\delta\boldsymbol{d}_i, \tag{6.39}$$

where, for node i, $\boldsymbol{d}_i$ is the vector of nodal variables,* $\delta\boldsymbol{d}_i$ is the vector of virtual nodal variables, $\boldsymbol{N}_i = \boldsymbol{I}\, N_i$ is the matrix of global shape functions† and $\boldsymbol{B}_i$ is the global strain–displacement matrix. The total number of nodes in the whole mesh is n.

If (6.38) and 6.39) are substituted into the virtual work expression (6.1) then we obtain

$$\sum_{i=1}^{n} [\delta\boldsymbol{d}_i]^T \left\{ \int_{\Omega} [\boldsymbol{B}_i]^T \boldsymbol{\sigma}\, d\Omega - \int_{\Omega} [\boldsymbol{N}_i]^T \boldsymbol{b}\, d\Omega - \int_{\Gamma_t} [\boldsymbol{N}_i]^T \boldsymbol{t}\, d\Gamma \right\} = 0, \tag{6.40}$$

and since (6.40) must be true for an arbitrary set of virtual displacements $\delta\boldsymbol{d}_i$ then we have for each node i an equation of the form

$$\int_{\Omega} [\boldsymbol{B}_i]^T \boldsymbol{\sigma}\, d\Omega - \int_{\Omega} [\boldsymbol{N}_i]^T \boldsymbol{b}\, d\Omega - \int_{\Gamma_t} [\boldsymbol{N}_i]^T \boldsymbol{t}\, d\Gamma = 0. \tag{6.41}$$

If we use $C(0)$ isoparametric finite element representations we can evaluate contributions to (6.41) separately from each element.

The displacements can be expressed in the usual way as

$$\boldsymbol{u}^{(e)} = \sum_{i=1}^{r} \boldsymbol{N}_i^{(e)} \boldsymbol{d}_i^{(e)}, \tag{6.42}$$

where, for local node i of element e, $\boldsymbol{N}^{(e)} = \boldsymbol{I}\, N^{(e)}$ is the matrix of shape functions and the vector of variables is $\boldsymbol{d}_i^{(e)}$. There are r local nodes in each element e.

Typical 4-, 8- and 9-node isoparametric element shape functions are shown and listed in Figs. 6.1(a), (b) and (c) respectively.

Note that in an isoparametric representation we may use the following representation for the x and y coordinates within an element

* In Part I of this text the nodal variables were symbolised by $\boldsymbol{\varphi}$; since for non-structural applications, such as nonlinear heat conduction, these parameters are not associated with displacements. In Parts II and III, for the continuum and plate situations considered, the nodal variables are always the displacement (and rotation) components and will now be symbolised by $\boldsymbol{d}$.

† Note that $\boldsymbol{I}$ is the $p \times p$ identity matrix in which $p=2$ for the plane stress, plane strain and axisymmetric applications and $p = 3$ for the Mindlin plate applications. N_i is the global shape function for node i.

$$\begin{bmatrix} x^{(e)} \\ y^{(e)} \end{bmatrix} = \sum_{i=1}^{r} \begin{bmatrix} N_i^{(e)} & 0 \\ 0 & N_i^{(e)} \end{bmatrix} \begin{bmatrix} x_i^{(e)} \\ y_i^{(e)} \end{bmatrix}, \tag{6.43}^*$$

in which $N_i^{(e)}$ are the same shape functions used in the displacement representation. We may then evaluate the Jacobian matrix as

$$J^{(e)} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \displaystyle\sum_{i=1}^{r} \frac{\partial N_i^{(e)}}{\partial \xi} x_i^{(e)} & \displaystyle\sum_{i=1}^{r} \frac{\partial N_i^{(e)}}{\partial \xi} y_i^{(e)} \\ \displaystyle\sum_{i=1}^{r} \frac{\partial N_i^{(e)}}{\partial \eta} x_i^{(e)} & \displaystyle\sum_{i=1}^{r} \frac{\partial N_i^{(e)}}{\partial \eta} y_i^{(e)} \end{bmatrix}. \tag{6.44}$$

The inverse of $J^{(e)}$ is then evaluated using the expression

$$[J^{(e)}]^{-1} = \begin{bmatrix} \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \eta}{\partial x} \\ \dfrac{\partial \xi}{\partial y} & \dfrac{\partial \eta}{\partial y} \end{bmatrix} = \frac{1}{\det J^{(e)}} \begin{bmatrix} \dfrac{\partial y}{\partial \eta} & -\dfrac{\partial y}{\partial \xi} \\ -\dfrac{\partial x}{\partial \eta} & \dfrac{\partial x}{\partial \xi} \end{bmatrix}. \tag{6.45}$$

The strain displacement relationships are expressed as

$$\boldsymbol{\epsilon}^{(e)} = \sum_{i=1}^{r} \boldsymbol{B}_i^{(e)} \boldsymbol{d}_i^{(e)}, \tag{6.46}$$

in which $\boldsymbol{B}_i^{(e)}$ is the strain matrix.

The discretised elemental volume (or area in the case of Mindlin plates) is given as

$$d\Omega^{(e)} = h^{(e)} \det \boldsymbol{J}^{(e)}\, d\xi\, d\eta, \tag{6.47}$$

where $h^{(e)}$ has been defined in Table 6.1 in which we also summarise the expressions for $\boldsymbol{d}_i^{(e)}$, $\boldsymbol{B}_i^{(e)}$ and $d\Omega^{(e)}$ for the four applications.

The Cartesian shape function derivatives used in the strain-displacement matrices in Table 6.1 may be obtained using the chain rule of differentiation

$$\frac{\partial N_i^{(e)}}{\partial x} = \frac{\partial N_i^{(e)}}{\partial \xi} \frac{\partial \xi}{\partial x} + \frac{\partial N_i^{(e)}}{\partial \eta} \frac{\partial \eta}{\partial x}, \tag{6.48}$$

* For axisymmetric problems replace x and y by r and z respectively.

Application	$d_i^{(e)}$	$B_i^{(e)}$	$d\Omega^{(e)}$
Plane stress	$\begin{bmatrix} u_i^{(e)} \\ v_i^{(e)} \end{bmatrix}$	$\begin{bmatrix} \left(\frac{\partial N_i}{\partial x}\right)^{(e)} & 0 \\ 0 & \left(\frac{\partial N_i}{\partial y}\right)^{(e)} \\ \left(\frac{\partial N_i}{\partial y}\right)^{(e)} & \left(\frac{\partial N_i}{\partial x}\right)^{(e)} \end{bmatrix}$	$t^{(e)} \det \boldsymbol{J}^{(e)} d\xi d\eta$
Plane strain	$\begin{bmatrix} u_i^{(e)} \\ v_i^{(e)} \end{bmatrix}$	$\begin{bmatrix} \left(\frac{\partial N_i}{\partial x}\right)^{(e)} & 0 \\ 0 & \left(\frac{\partial N_i}{\partial y}\right)^{(e)} \\ \left(\frac{\partial N_i}{\partial y}\right)^{(e)} & \left(\frac{\partial N_i}{\partial x}\right)^{(e)} \end{bmatrix}$	$\det \boldsymbol{J}^{(e)} d\xi d\eta$
Axial symmetry	$\begin{bmatrix} u_i^{(e)} \\ w_i^{(e)} \end{bmatrix}$	$\begin{bmatrix} \left(\frac{\partial N_i}{\partial r}\right)^{(e)} & 0 \\ \left(\frac{N_i}{r}\right)^{(e)} & 0 \\ 0 & \left(\frac{\partial N_i}{\partial z}\right)^{(e)} \\ \left(\frac{\partial N_i}{\partial z}\right)^{(e)} & \left(\frac{\partial N_i}{\partial r}\right)^{(e)} \end{bmatrix}$	$2\pi r^{(e)} \det \boldsymbol{J}^{(e)} d\xi d\eta$
Mindlin plate	$\begin{bmatrix} w_i^{(e)} \\ \theta_{xi}^{(e)} \\ \theta_{yi}^{(e)} \end{bmatrix}$	$\begin{bmatrix} 0 & \left(-\frac{\partial N_i}{\partial x}\right)^{(e)} & 0 \\ 0 & 0 & \left(-\frac{\partial N_i}{\partial y}\right)^{(e)} \\ 0 & \left(-\frac{\partial N_i}{\partial y}\right)^{(e)} & \left(-\frac{\partial N_i}{\partial x}\right)^{(e)} \\ \left(\frac{\partial N_i}{\partial x}\right)^{(e)} & -N_i^{(e)} & 0 \\ \left(\frac{\partial N_i}{\partial y}\right)^{(e)} & 0 & -N_i^{(e)} \end{bmatrix}$	$\det \boldsymbol{J}^{(e)} d\xi d\eta$

Table 6.1 Nodal displacements, strain matrices and elemental volumes or areas for two-dimensional solid mechanics applications.

and

$$\frac{\partial N_i^{(e)}}{\partial y} = \frac{\partial N_i^{(e)}}{\partial \eta}\frac{\partial \eta}{\partial y} + \frac{\partial N_i^{(e)}}{\partial \xi}\frac{\partial \xi}{\partial y},$$

in which the terms $\partial\xi/\partial x$, $\partial\eta/\partial x$, $\partial\eta/\partial y$ and $\partial\xi/\partial y$ may be obtained from the inverse of the Jacobian matrix given in (6.45).

Since we have a linear stress–strain relationship within each element of the form

$$\boldsymbol{\sigma}^{(e)} = \boldsymbol{D}^{(e)}\boldsymbol{\epsilon}^{(e)} = \boldsymbol{D}^{(e)}\left(\sum_{j=1}^{r} \boldsymbol{B}_j^{(e)}\boldsymbol{d}_j^{(e)}\right), \qquad (6.49)$$

then the contribution from element e to the first term in (6.41) is given as

$$\sum_{j=1}^{r} \boldsymbol{K}_{ij}^{(e)}\boldsymbol{d}_j^{(e)} \equiv \int_{\Omega^{(e)}} [\boldsymbol{B}_i^{(e)}]^T \boldsymbol{D}^{(e)}\left(\sum_{j-1}^{r} \boldsymbol{B}_j^{(e)}\boldsymbol{d}_j^{(e)}\right) d\Omega, \qquad (6.50)$$

where $\boldsymbol{K}_{ij}^{(e)}$ is the submatrix of element stiffness matrix $\boldsymbol{K}^{(e)}$.

The contribution from element e to the second term in (6.41) is given as

$$\boldsymbol{f}_{B_i}^{(e)} = \int_{\Omega^{(e)}} [N_i^{(e)}]^T \boldsymbol{b}^{(e)} d\Omega. \qquad (6.51)$$

For the third term, the contribution from element e is

$$\boldsymbol{f}_{T_i}^{(e)} = \int_{\Gamma_t^{(e)}} [N_i^{(e)}]^T \boldsymbol{t}^{(e)} d\Gamma, \qquad (6.52)$$

where $\Gamma_t^{(e)}$ is that part of Γ_t which coincides with a boundary of element e. Of course for many elements there will be no contribution to $\boldsymbol{f}_{Ti}^{(e)}$.

6.3.2 Evaluation of the stiffness matrix and consistent load vector

Let us now consider the evaluation of $\boldsymbol{K}$.

The integration is now performed in the natural coordinate system. Thus the submatrix of the stiffness matrix $\boldsymbol{K}^{(e)}$ linking nodes i and j has the form

$$\boldsymbol{K}_{ij}^{(e)} = \int_{-1}^{+1}\int_{-1}^{+1} [\boldsymbol{B}_i^{(e)}]^T \boldsymbol{D}^{(e)}\boldsymbol{B}_j^{(e)} h^{(e)} \det \boldsymbol{J}^{(e)} d\xi\, d\eta. \qquad (6.53)$$

The elements of $\boldsymbol{K}_{ij}^{(e)}$ are evaluated numerically. If the integrand in (6.53) is denoted as

$$[\boldsymbol{B}_i^{(e)}]^T \boldsymbol{D}^{(e)} \boldsymbol{B}_j^{(e)} h^{(e)} \det \boldsymbol{J}^{(e)} = \boldsymbol{T}_{ij}^{(e)}, \qquad (6.54)$$

then

$$\boldsymbol{K}_{ij}^{(e)} = \int_{-1}^{+1}\int_{-1}^{+1} \boldsymbol{T}_{ij}^{(e)} d\xi\, d\eta. \qquad (6.55)$$

The numerical integration for a quadrilateral element with $n \times n$ sampling points leads to

$$K_{ij}^{(e)} = \sum_{p=1}^{n} \sum_{q=1}^{n} T(\xi_p, \bar{\eta}_q)_{ij} W_p W_q, \tag{6.56}$$

where W_p and W_q are weighting factors and $(\xi_p, \bar{\eta}_q)$ is a sampling position.

The consistent nodal forces at node i caused by body forces are

$$f_{B_i}^{(e)} = \int_{-1}^{+1} \int_{-1}^{+1} [N_i^{(e)}]^T b^{(e)} h^{(e)} \det J^{(e)} d\xi d\eta. \tag{6.57}$$

The components of $f_{Bi}^{(e)}$ are evaluated numerically. If the integrand in (6.57) is denoted as

$$g_i^{(e)} = [N_i^{(e)}]^T b^{(e)} h^{(e)} \det J^{(e)}, \tag{6.58}$$

then

$$f_{B_i}^{(e)} = \int_{-1}^{+1} \int_{-1}^{+1} g_i^{(e)} d\xi d\eta. \tag{6.59}$$

The numerical integration for a quadrilateral with $n \times n$ sampling points leads to

$$f_{B_i}^{(e)} = \sum_{p=1}^{n} \sum_{q=1}^{n} g(\xi_p, \bar{\eta}_q)_i^{(e)} W_p W_q, \tag{6.60}$$

where W_p and W_q are weighting factors and $(\xi_p, \bar{\eta}_q)$ is a sampling position.

The consistent nodal forces for boundary tractions have been dealt with in the authors' previous book[4] and will be summarised in Section 6.4.5.

The computer implementation of numerically integrated isoparametric elements has been described in detail in the text of *Finite Element Programming*.[4] Here we simply summarise in Fig. 6.6 the main steps involved in evaluating the element stiffness matrix.

6.4 Standard subroutines for linear elastic finite element analysis

Many of the subroutines required for elasto-plastic finite element analysis are common to the corresponding linear elastic application. In this section we present all the standard linear elastic subroutines required for later use in Chapters 7, 8 and 9. The function of each subroutine is explained and a FORTRAN listing is provided. The subroutines presented are drawn from Ref. 4 where a detailed description is provided.

In order to make all subroutines modular in form we have adopted a type of dynamic dimensioning. Thus no COMMON blocks are used in the programs in Part II. Dimensions are fixed in the main or master routine and all necessary information is transmitted between routines by the use of

SUBROUTINE STIF2D
Dimensions and common blocks.
→ Enter loop over all elements.
Retrieve element geometry and material properties for the current element.
Zero the stiffness array.
Call a routine which sets up $\boldsymbol{D}^{(e)}$ the constitutive matrix.
→ Enter loops covering all integration points.
Look up sampling position for the current integration point $(\bar{\xi}_p, \bar{\eta}_q)$.
Call shape function routine SFR2—given $(\bar{\xi}_p, \bar{\eta}_q)$ this will return the shape functions $N_i^{(e)}$ and their derivatives $\partial N_i^{(e)}/\partial \xi$ and $\partial N_i^{(e)}/\partial \eta$ at the point $(\bar{\xi}_p, \bar{\eta}_q)$.
Call JACOB2—given $N_i^{(e)}$, $\partial N_i^{(e)}/\partial \xi$ and $\partial N_i^{(e)}/\partial \eta$ at point $(\bar{\xi}_p, \bar{\eta}_q)$; this will return Cartesian shape function derivatives $\partial N_i^{(e)}/\partial x$ and $\partial N_i^{(e)}/\partial y$, the Jacobian matrix $\boldsymbol{J}^{(e)}$, its inverse $[\boldsymbol{J}^{(e)}]^{-1}$ and its determinant det $\boldsymbol{J}^{(e)}$ and the x and y (or r and z) coordinates all at the point $(\bar{\xi}_p, \bar{\eta}_q)$.
Call strain matrix routine—given $N_i^{(e)}$, $\partial N_i^{(e)}/\partial x$ and $\partial N_i^{(e)}/\partial y$ at $(\bar{\xi}_p, \bar{\eta}_q)$ this will return the strain matrix $\boldsymbol{B}_i^{(e)}$.
Call a routine to evaluate $\boldsymbol{D}^{(e)} \boldsymbol{B}^{(e)}$.
Evaluate $[\boldsymbol{B}_i^{(e)}]\boldsymbol{D}^{(e)} \boldsymbol{B}_j^{(e)} \det \boldsymbol{J}^{(e)} \times$ integration weights and assemble them into the element stiffness array $\boldsymbol{K}_{ij}^{(e)}$.
Assemble $\boldsymbol{D}^{(e)} \boldsymbol{B}^{(e)}$ into a stress array for later evaluation of stresses from the nodal displacements.
End integration loops.
Write stiffness matrix and stress matrix onto file for use in the solution routine.
End element loop.
RETURN END

Fig. 6.6 Evaluation of element stiffness matrices for numerically integrated isoparametric elements.

arguments (and also peripherals in certain instances). Apart from the modularity, this approach has the advantage that maximum dimensions can be updated in a very simple and straightforward manner. Only the DIMENSION statement in the main segment and some statements in a subroutine which sets the maximum dimensions sizes need modification.

As an example, the relevant statements in a dynamically dimensioned program are listed below.

```
PROGRAM      FRED (      )
DIMENSION    AMATX (200, 5), ...*
   .
   .
   .
CALL       DIMENS (MROWS, MCOLS)
   .
   .
   .
CALL       DUMMY (AMATX, MROWS, MCOLS)
   .
   .
   .
STOP
END

SUBROUTINE   DIMENS (MROWS, MCOLS)
MROWS      =200*
MCOLS      = 5*
RETURN
END

SUBROUTINE   DUMMY (AMATX, MROWS, MCOLS)
DIMENSION    AMATX (MROWS, MCOLS)
   .
   .
   .
RETURN
END
```

Note that AMATX ( ) has fixed dimensions in the main routine FRED. Subroutine DIMENS assigns values of 200 and 5 to the dimensions MROWS and MCOLS respectively.† In subroutine DUMMY we transmit AMATX,

† Alternatively a DATA statement can be used.

MROWS and MCOLS via the argument and therefore the DIMENSION statement in DUMMY refers to AMATX (MCOLS, MROWS) and not AMATX (200, 5). To update FRED for arrays AMATX with different maximum dimensions, we simply modify those statements indicated by an asterisk.

Note also that the use of such arguments is not very expensive since only the address of the first term of an array is passed through the argument and not of all the terms in the array.

More sophisticated versions of this approach can be implemented as illustrated in the book by Irons and Ahmad.[5] Such approaches undoubtedly save core storage but they do require careful housekeeping and checking procedures.

In Part III we have generally dispensed with the use of maximum dimension variables in the programs. Thus main segment FRED would then be written as

```
PROGRAM  FRED (       )
DIMENSION     AMATX (200, 5), . . .
   .
   .
   .
CALL          DUMMY (AMATX)
   .
   .
   .
STOP
END

SUBROUTINE    DUMMY (AMATX)
DIMENSION     AMATX (200, 1)†
   .
   .
   .
RETURN
END
```

Although this approach uses nonstandard FORTRAN IV it does work on most machines and it has been adopted elsewhere in the literature.[6] If more than one subroutine such as DUMMY uses AMATX then the relevant dimensions must be identical in all of these subroutines.

The list of variables in the argument list will differ between linear and nonlinear applications. For each subroutine presented in this section the form of the argument list and the dimension statements will be those required for two-dimensional elasto-plastic applications.

† Note that AMATX (*number*, 1) will also workprovided that *number* $\leqslant$ 200.

6.4.1 Subroutine NODEXY for generating coordinate values for midside nodes

For the quadratic 8- and 9-node elements described in Section 6.3 subroutine NODEXY checks each midside node (a midside node being recognisable from the element topology cards). If both coordinates of a midside node are found to be zero, its coordinates are linearly interpolated between the two adjacent corner nodes. Subroutine NODEXY is common to plane stress/strain, axisymmetric and plate bending situations.

```
      SUBROUTINE NODEXY(COORD,LNODS,MELEM,MPOIN,NELEM,NNODE)
C***********************************************************************
C
C**** THIS SUBROUTINE INTERPOLATES THE MIDE SIDE NODES OF STRAIGHT
C     SIDES OF ELEMENTS AND THE CENTRAL NODE OF 9 NODED ELEMENTS
C
C***********************************************************************
      DIMENSION COORD(MPOIN,2),LNODS(MELEM,9)
      IF(NNODE.EQ.4) RETURN
C
C*** LOOP OVER EACH ELEMENT
C
      DO 30 IELEM=1,NELEM
C
C*** LOOP OVER EACH ELEMENT EDGE
C
      NNOD1=9
      IF(NNODE.EQ.8) NNOD1=7
      DO 20 INODE=1,NNOD1,2
      IF(INODE.EQ.9) GO TO 50
C
C*** COMPUTE THE NODE NUMBER OF THE FIRST NODE
C
      NODST=LNODS(IELEM,INODE)
      IGASH=INODE+2
      IF(IGASH.GT.8) IGASH=1
C
C*** COMPUTE THE NODE NUMBER OF THE LAST NODE
C
      NODFN=LNODS(IELEM,IGASH)
      MIDPT=INODE+1
C
C*** COMPUTE THE NODE NUMBER OF THE INTERMEDIATE NODE
C
      NODMD=LNODS(IELEM,MIDPT)
      TOTAL=ABS(COORD(NODMD,1))+ABS(COORD(NODMD,2))
C
C*** IF THE COORDINATES OF THE INTERMEDIATE NODE ARE BOTH ZERO
C    INTERPOLATE BY A STRAIGHT LINE
C
      IF(TOTAL.GT.0.0) GO TO 20
      KOUNT=1
   10 COORD(NODMD,KOUNT)=(COORD(NODST,KOUNT)+COORD(NODFN,KOUNT))/2.0
      KOUNT=KOUNT+1
      IF(KOUNT.EQ.2) GO TO 10
   20 CONTINUE
      GO TO 30
   50 LNODE=LNODS(IELEM,INODE)
      TOTAL=ABS(COORD(LNODE,1))+ABS(COORD(LNODE,2))
      IF(TOTAL.GT.0.0) GO TO 30
```

```
      LNOD1=LNODS(IELEM,1)                                              NODE  51
      LNOD3=LNODS(IELEM,3)                                              NODE  52
      LNOD5=LNODS(IELEM,5)                                              NODE  53
      LNOD7=LNODS(IELEM,7)                                              NODE  54
      KOUNT=1                                                           NODE  55
   40 COORD(LNODE,KOUNT)=(COORD(LNOD1,KOUNT)+COORD(LNOD3,KOUNT)         NODE  56
     . +COORD(LNOD5,KOUNT)+COORD(LNOD7,KOUNT))/4.0                      NODE  57
      KOUNT=KOUNT+1                                                     NODE  58
      IF(KOUNT.EQ.2) GO TO 40                                           NODE  59
   30 CONTINUE                                                          NODE  60
      RETURN                                                            NODE  61
      END                                                               NODE  62
```

6.4.2 Subroutine GAUSSQ for generating Gaussian quadrature data

The function of this subroutine is to set up the sampling point positions and weighting factors for numerical integration. The Gauss quadrature processes utilised in this text are restricted to either two or three point integration rules.* The role of numerical integration in the isoparametric formulation was discussed in detail in Section 6.3. The order of integration rule to be employed is defined by NGAUS and the sampling point positions and weighting factors are stored respectively in arrays POSGP( ) and WEIGP( ).

```
      SUBROUTINE GAUSSQ(NGAUS,POSGP,WEIGP)                              GAUS   1
C*********************************************************************  GAUS   2
C                                                                       GAUS   3
C**** THIS SUBROUTINE SETS UP THE GAUSS-LEGENDRE INTEGRATION CONSTANTS  GAUS   4
C                                                                       GAUS   5
C*********************************************************************  GAUS   6
      DIMENSION POSGP(4),WEIGP(4)                                       GAUS   7
      IF(NGAUS.GT.2) GO TO 4                                            GAUS   8
    2 POSGP(1)=-0.577350269189626                                       GAUS   9
      WEIGP(1)=1.0                                                      GAUS  10
      GO TO 6                                                           GAUS  11
    4 POSGP(1)=-0.774596669241483                                       GAUS  12
      POSGP(2)=0.0                                                      GAUS  13
      WEIGP(1)=0.555555555555556                                        GAUS  14
      WEIGP(2)=0.888888888888889                                        GAUS  15
    6 KGAUS=NGAUS/2                                                     GAUS  16
      DO 8 IGASH=1,KGAUS                                                GAUS  17
      JGASH=NGAUS+1-IGASH                                               GAUS  18
      POSGP(JGASH)=-POSGP(IGASH)                                        GAUS  19
      WEIGP(JGASH)=WEIGP(IGASH)                                         GAUS  20
    8 CONTINUE                                                          GAUS  21
      RETURN                                                            GAUS  22
      END                                                               GAUS  23
```

6.4.3 Subroutine SFR2 for evaluating the element shape functions

The role of this subroutine is to evaluate the shape functions $N_i^{(e)}(\xi, \eta)$ and their derivatives $\partial N_i^{(e)}/\partial \xi$, $\partial N_i^{(e)}/\partial \eta$ at any sampling point ξ_P, η_P within the element for each of the 4-, 8- or 9-noded elements described in Section 6.1. The shape functions for these elements are listed in Figs. 6.1(a), (b) and (c). The sampling point coordinates ξ_P, η_P are specified as EXISP and ETASP respectively. The evaluated shape functions for each node of an element are stored in array SHAPE (INODE) and their derivatives in

* Except for selectively integrated 4-node Mindlin plates in which we modify GAUSSQ so that if NGAUS = 1 then POSGP(1) = 0·0 and WEIGP(1) = 2·0.

array DERIV (INODE, IDIME) where INODE ranges over the element nodes and IDIME over the coordinate dimensions.

```
      SUBROUTINE SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
C**********************************************************************
C
C**** THIS SUBROUTINE EVALUATES SHAPE FUNCTIONS AND THEIR DERIVATIVES
C     FOR LINEAR,QUADRATIC LAGRANGIAN AND SERENDIPITY
C     ISOPARAMETRIC 2-D ELEMENTS
C
C**********************************************************************
      DIMENSION DERIV(2,9) ,SHAPE(9)
      S=EXISP
      T=ETASP
      IF(NNODE.GT.4) GO TO 10
      ST=S*T
C
C*** SHAPE FUNCTIONS FOR 4 NODED ELEMENT
C
      SHAPE(1)=(1-T-S+ST)*0.25
      SHAPE(2)=(1-T+S-ST)*0.25
      SHAPE(3)=(1+T+S+ST)*0.25
      SHAPE(4)=(1+T-S-ST)*0.25
C
C*** SHAPE FUNCTION DERIVATIVES
C
      DERIV(1,1)=(-1+T)*0.25
      DERIV(1,2)=(+1-T)*0.25
      DERIV(1,3)=(+1+T)*0.25
      DERIV(1,4)=(-1-T)*0.25
      DERIV(2,1)=(-1+S)*0.25
      DERIV(2,2)=(-1-S)*0.25
      DERIV(2,3)=(+1+S)*0.25
      DERIV(2,4)=(+1-S)*0.25
      RETURN
   10 IF(NNODE.GT.8)GO TO 30
      S2=S*2.0
      T2=T*2.0
      SS=S*S
      TT=T*T
      ST=S*T
      SST=S*S*T
      STT=S*T*T
      ST2=S*T*2.0
C
C*** SHAPE FUNCTIONS FOR 8 NODED ELEMENT
C
      SHAPE(1)=(-1.0+ST+SS+TT-SST-STT)/4.0
      SHAPE(2)=(1.0-T-SS+SST)/2.0
      SHAPE(3)=(-1.0-ST+SS+TT-SST+STT)/4.0
      SHAPE(4)=(1.0+S-TT-STT)/2.0
      SHAPE(5)=(-1.0+ST+SS+TT+SST+STT)/4.0
      SHAPE(6)=(1.0+T-SS-SST)/2.0
      SHAPE(7)=(-1.0-ST+SS+TT+SST-STT)/4.0
      SHAPE(8)=(1.0-S-TT+STT)/2.0
C*** SHAPE FUNCTION DERIVATIVES
C
      DERIV(1,1)=(T+S2-ST2-TT)/4.0
      DERIV(1,2)=-S+ST
      DERIV(1,3)=(-T+S2-ST2+TT)/4.0
      DERIV(1,4)=(1.0-TT)/2.0
      DERIV(1,5)=(T+S2+ST2+TT)/4.0
      DERIV(1,6)=-S-ST
```

```
      DERIV(1,7)=(-T+S2+ST2-TT)/4.0
      DERIV(1,8)=(-1.0+TT)/2.0
      DERIV(2,1)=(S+T2-SS-ST2)/4.0
      DERIV(2,2)=(-1.0+SS)/2.0
      DERIV(2,3)=(-S+T2-SS+ST2)/4.0
      DERIV(2,4)=-T-ST
      DERIV(2,5)=(S+T2+SS+ST2)/4.0
      DERIV(2,6)=(1.0-SS)/2.0
      DERIV(2,7)=(-S+T2+SS-ST2)/4.0
      DERIV(2,8)=-T+ST
      RETURN
   30 CONTINUE
      SS=S*S
      ST=S*T
      TT=T*T
      S1=S+1.0
      T1=T+1.0
      S2=S*2.0
      T2=T*2.0
      S9=S-1.0
      T9=T-1.0
C
C*** SHAPE FUNCTIONS FOR 9 NODED ELEMENT
C
      SHAPE(1)=0.25*S9*ST*T9
      SHAPE(2)=0.5*(1.0-SS)*T*T9
      SHAPE(3)=0.25*S1*ST*T9
      SHAPE(4)=0.5*S*S1*(1.0-TT)
      SHAPE(5)=0.25*S1*ST*T1
      SHAPE(6)=0.5*(1.0-SS)*T*T1
      SHAPE(7)=0.25*S9*ST*T1
      SHAPE(8)=0.5*S*S9*(1.0-TT)
      SHAPE(9)=(1.0-SS)*(1.0-TT)
C
C*** SHAPE FUNCTION DERIVATIVES
C
      DERIV(1,1)=0.25*T*T9*(-1.0+S2)
      DERIV(1,2)=-ST*T9
      DERIV(1,3)=0.25*(1.0+S2)*T*T9
      DERIV(1,4)=0.5*(1.0+S2)*(1.0-TT)
      DERIV(1,5)=0.25*(1.0+S2)*T*T1
      DERIV(1,6)=-ST*T1
      DERIV(1,7)=0.25*(-1.0+S2)*T*T1
      DERIV(1,8)=0.5*(-1.0+S2)*(1.0-TT)
      DERIV(1,9)=-S2*(1.0-TT)
      DERIV(2,1)=0.25*(-1.0+T2)*S*S9
      DERIV(2,2)=0.5*(1.0-SS)*(-1.0+T2)
      DERIV(2,3)=0.25*S*S1*(-1.0+T2)
      DERIV(2,4)=-ST*S1
      DERIV(2,5)=0.25*S*S1*(1.0+T2)
      DERIV(2,6)=0.5*(1.0-SS)*(1.0+T2)
      DERIV(2,7)=0.25*S*S9*(1.0+T2)
      DERIV(2,8)=-ST*S9
      DERIV(2,9)=-T2*(1.0-SS)
   20 CONTINUE
      RETURN
      END
```

6.4.4 Subroutine JACOB2 for evaluating the Jacobian matrix

This subroutine calculates, for any sampling position, ξ_P, η_P (usually the Gauss point), the following quantities:

- The Cartesian coordinates of the Gauss point which are stored in the array GPCOD ( ).
- The Jacobian matrix which is stored in XJACM ( ). For two-dimensional applications the Jacobian matrix is defined by (6.44).
- The determinant of the Jacobian matrix, DJACB.
- The inverse of the Jacobian matrix which is stored as XJACI ( ).
- The Cartesian derivatives $\partial N_i^{(e)}/\partial x$, $\partial N_i^{(e)}/\partial y$ (or $\partial N_i^{(e)}/\partial r$, $\partial N_i^{(e)}/\partial z$), of the element shape functions. These quantities are defined in (6.48).

```
      SUBROUTINE JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .                  NNODE,SHAPE)
C*********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE JACOBIAN MATRIX AND THE CARTESIAN
C     SHAPE FUNCTION DERIVATIVES
C
C*********************************************************************
      DIMENSION CARTD(2,9),DERIV(2,9),ELCOD(2,9) ,GPCOD(2,9) ,SHAPE(9),
     .          XJACI(2,2),XJACM(2,2)
C
C***  CALCULATE COORDINATES OF SAMPLING POINT
C
      DO 2 IDIME=1,2
      GPCOD(IDIME,KGASP)=0.0
      DO 2 INODE=1,NNODE
      GPCOD(IDIME,KGASP)=GPCOD(IDIME,KGASP)+ELCOD(IDIME,INODE)
     .*SHAPE(INODE)
    2 CONTINUE
C
C***  CREATE JACOBIAN MATRIX XJACM
C
      DO 4 IDIME=1,2
      DO 4 JDIME=1,2
      XJACM(IDIME,JDIME)=0.0
      DO 4 INODE=1,NNODE
      XJACM(IDIME,JDIME)=XJACM(IDIME,JDIME)+DERIV(IDIME,INODE)*
     .ELCOD(JDIME,INODE)
    4 CONTINUE
C
C***  CALCULATE DETERMINANT AND INVERSE OF JACOBIAN MATRIX
C
      DJACB=XJACM(1,1)*XJACM(2,2)-XJACM(1,2)*XJACM(2,1)
      IF(DJACB) 6,6,8
    6 WRITE(6,600) IELEM
      STOP
    8 CONTINUE
      XJACI(1,1)=XJACM(2,2)/DJACB
      XJACI(2,2)=XJACM(1,1)/DJACB
      XJACI(1,2)=-XJACM(1,2)/DJACB
      XJACI(2,1)=-XJACM(2,1)/DJACB
C
C***  CALCULATE CARTESIAN DERIVATIVES
C
      DO 10 IDIME=1,2
      DO 10 INODE=1,NNODE
      CARTD(IDIME,INODE)=0.0
      DO 10 JDIME=1,2
      CARTD(IDIME,INODE)=CARTD(IDIME,INODE)+XJACI(IDIME,JDIME)*
     .DERIV(JDIME,INODE)
```

```
 10 CONTINUE
600 FORMAT(//,36H PROGRAM HALTED IN SUBROUTINE JACOB2,/,11X,
   .22H ZERO OR NEGATIVE AREA,/,10X,16H ELEMENT NUMBER ,I5)
    RETURN
    END
```

6.4.5 Subroutine LOADPS for evaluating the element nodal forces for plane and axisymmetric situations

The role of this subroutine is to evaluate the consistent nodal forces for each element due to discrete point loads, gravity loading and distributed edge loading/unit length of element. This subroutine is described in detail in Chapter 7, Ref. 4. The types of loading to be considered are controlled by input parameters IPLOD, IGRAV, IEDGE. Nonzero values of these respective items indicate that point loads, gravity loading or distributed edge loading is to be considered.

The consistent nodal loads are evaluated for each element separately and stored in the array RLOAD (IELEM, IEVAB) where IELEM indicates the element and IEVAB ranges over the degrees of freedom of the element. For equation solution by the *frontal process* it is not necessary to evaluate the total applied load acting at each node, with instead each element contribution being assembled directly into the global load vector during equation assembly and solution.

Point loads

If parameter IPLOD is nonzero the applied nodal loads are read as input. For each particular node the applied forces are associated with any one of the elements attached to it; since each element contribution will be assembled before equation solution. Thus a search is performed over all elements until the node number is found in an element and the nodal loads are then associated with the appropriate degrees of freedom of that element.

Gravity loading

For plane stress or plane strain problems the direction in which gravity acts need not coincide with either of the coordinate axes. Therefore the direction in which gravity acts must be defined as shown in Fig. 6.7 by specifying the angle θ which the gravity axis makes with the positive y axis. The intensity of the loading is defined by specifying the gravitational acceleration, g, which acts. For axisymmetric problems, of course, the gravity axis must coincide with the z axis.

The consistent nodal forces for node i of an element are then given by

$$\begin{bmatrix} P_{xi} \\ P_{yi} \end{bmatrix}^{(e)} = \int_{\Omega^{(e)}} N_i^{(e)} \rho g \begin{bmatrix} \sin\theta \\ -\cos\theta \end{bmatrix} d\Omega, \tag{6.61}$$

in which ρ is the material mass density. Integrated numerically this becomes

$$\begin{bmatrix} P_{xi} \\ P_{yi} \end{bmatrix}^{(e)} = \sum_{n=1}^{NGAUS} \sum_{m=1}^{NGAUS} \rho g t \begin{bmatrix} \sin\theta \\ -\cos\theta \end{bmatrix} N_i(\xi_n, \eta_m) W_n W_m \det \boldsymbol{J}, \qquad (6.62)$$

where t is the element thickness for plane problems. For axisymmetric applications t is replaced by $2\pi r_P$, where r_P is the radial distance to the Gauss point under consideration.

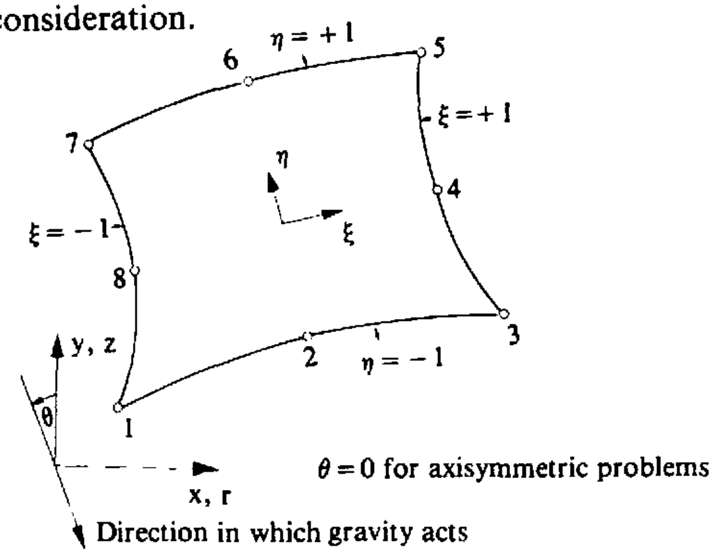


Fig. 6.7 Specification of the gravity axis for two-dimensional problems.

Distributed edge loading

Any element edge can have a distributed loading per unit length in a normal and tangential direction prescribed to it as shown in Fig. 6.8. These distributed forces can vary (independently) along the edges. For the quadratic elements considered in this text, a quadratic loading distribution can, at best, be accommodated. The variation is defined by prescribing the normal and tangential values at the three nodal points forming the element edge to which the loads are applied. For linear quadrilateral elements, only a linear distributed load variation can be accommodated. In order to be consistent with the order of listing of nodal connection numbers in the element topology definition, the three (or two) nodes forming the loaded edge must also be listed in an anticlockwise sequence with respect to the loaded element. The positive directions of normal and tangential loading are indicated in Fig. 6.8.

The consistent nodal forces for node i can be shown to be[4]

$$P_{xi}^{(e)} = \int_{\Gamma^{(e)}} N_i^{(e)} \left(p_t \frac{\partial x}{\partial \xi} - p_n \frac{\partial y}{\partial \xi} \right) d\xi$$

$$P_{yi}^{(e)} = \int_{\Gamma^{(e)}} N_i^{(e)} \left(p_n \frac{\partial x}{\partial \xi} + p_t \frac{\partial y}{\partial \xi} \right) d\xi, \qquad (6.63)$$

where p_n and p_t are the normal and tangential distributed loads respectively. Integration is taken along the loaded element edge $\Gamma^{(e)}$, which is arbitrarily chosen to be defined by $\eta = -1$, as shown in Fig. 6.8.

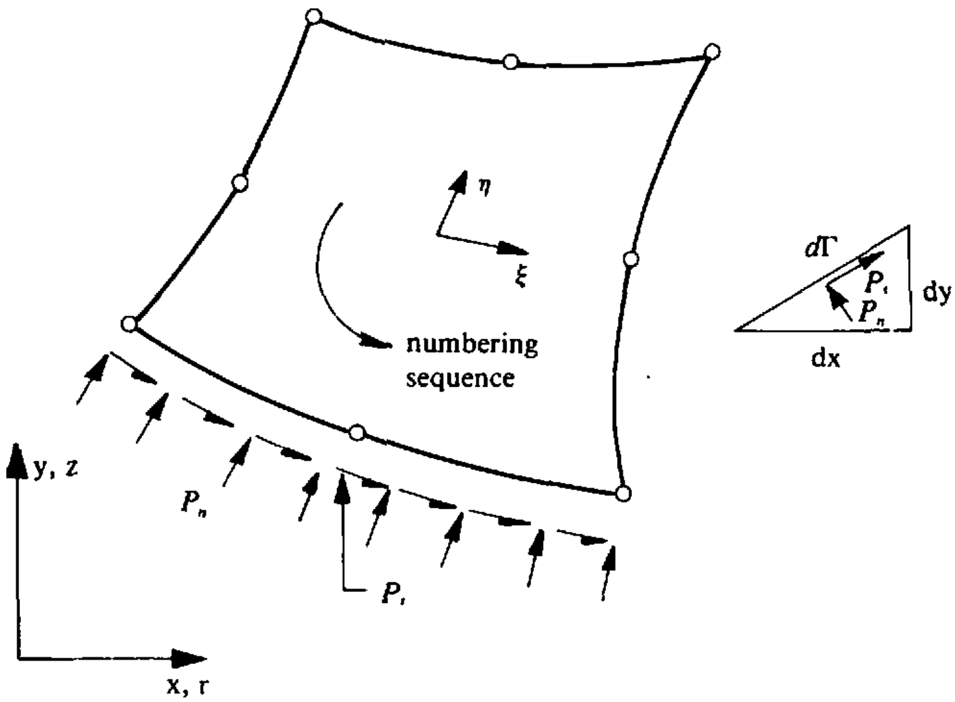


Fig. 6.8 Normal and tangential distributed loading on an element edge.

For axisymmetric problems the edge loading is in fact a distributed loading/unit area, since integration is additionally made over the circumferential direction.

If more than one type of loading acts on an element, the total nodal forces are accumulated and stored in array RLOAD. This total loading is then applied incrementally during elasto-plastic solution.

```
      SUBROUTINE LOADPS(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
     .                  NEVAB,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
     .                  PROPS,RLOAD,WEIGP,NDOFN)
     .
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE CONSISTENT NODAL FORCES FOR EACH
C     ELEMENT
C
C***********************************************************************
      DIMENSION CARTD(2,9) ,COORD(MPOIN,2),DERIV(2,9) ,DGASH(2),
     .          DMATX(4,4),ELCOD(2,9),LNODS(MELEM,9),MATNO(MELEM),
     .          NOPRS(4),PGASH(2),POINT(2),POSGP(4),PRESS(4,2),
     .          PROPS(MMATS,7),RLOAD(MELEM,18),SHAPE(9),STRAN(4),
     .          STRES(4),TITLE(12),
     .          WEIGP(4),GPCOD(2,9)
      TWOPI=6.283185308
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 RLOAD(IELEM,IEVAB)=0.0
      READ(5,901) TITLE
  901 FORMAT(12A6)
      WRITE(6,903) TITLE
  903 FORMAT(1H0,12A6)
```

```
C
C*** READ DATA CONTROLLING LOADING TYPES TO BE INPUTTED
C
      READ(5,919) IPLOD,IGRAV,IEDGE
      WRITE(6,919) IPLOD,IGRAV,IEDGE
  919 FORMAT(3I5)
C
C*** READ NODAL POINT LOADS
C
      IF(IPLOD.EQ.0) GO TO 500
   20 READ(5,931) LODPT,(POINT(IDOFN),IDOFN=1,2)
      WRITE(6,931) LODPT,(POINT(IDOFN),IDOFN=1,2)
  931 FORMAT(I5,2F10.3)
C
C*** ASSOCIATE THE NODAL POINT LOADS WITH AN ELEMENT
C
      DO 30 IELEM=1,NELEM
      DO 30 INODE=1,NNODE
      NLOCA=IABS(LNODS(IELEM,INODE))
   30 IF(LODPT.EQ.NLOCA) GO TO 40
   40 DO 50 IDOFN=1,2
      NGASH=(INODE-1)*2+IDOFN
   50 RLOAD(IELEM,NGASH)=POINT(IDOFN)
      IF(LODPT.LT.NPOIN) GO TO 20
  500 CONTINUE
      IF(IGRAV.EQ.0) GO TO 600
C
C*** GRAVITY LOADING SECTION
C
C
C*** READ GRAVITY ANGLE AND GRAVITATIONAL CONSTANT
C
      READ(5,906) THETA,GRAVY
  906 FORMAT(2F10.3)
      WRITE(6,911) THETA,GRAVY
  911 FORMAT(1H0,16H GRAVITY ANGLE =,F10.3,19H GRAVITY CONSTANT =,F10.3)
      THETA=THETA/57.295779514
C
C*** LOOP OVER EACH ELEMENT
C
      DO 90 IELEM=1,NELEM
C
C*** SET UP PRELIMINARY CONSTANTS
C
      LPROP=MATNO(IELEM)
      THICK=PROPS(LPROP,3)
      DENSE=PROPS(LPROP,4)
      IF(DENSE.EQ.0.0) GO TO 90
      GXCOM=DENSE*GRAVY*SIN(THETA)
      GYCOM=-DENSE*GRAVY*COS(THETA)
C
C*** COMPUTE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 60 INODE=1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      DO 60 IDIME=1,2
   60 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      KGASP=0
      DO 80 IGAUS=1,NGAUS
      DO 80 JGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
```

```
C
C*** COMPUTE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS AND ELEMENTAL
C    VOLUME
C
      CALL       SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      KGASP=KGASP+1
      CALL       JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .                  NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(THICK.NE.0.0) DVOLU=DVOLU*THICK
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
C
C*** CALCULATE LOADS AND ASSOCIATE WITH ELEMENT NODAL POINTS
C
      DO 70 INODE=1,NNODE
      NGASH=(INODE-1)*2+1
      MGASH=(INODE-1)*2+2
      RLOAD(IELEM,NGASH)=RLOAD(IELEM,NGASH)+GXCOM*SHAPE(INODE)*DVOLU
   70 RLOAD(IELEM,MGASH)=RLOAD(IELEM,MGASH)+GYCOM*SHAPE(INODE)*DVOLU
   80 CONTINUE
   90 CONTINUE
  600 CONTINUE
      IF(IEDGE.EQ.0) GO TO 700
C
C*** DISTRIBUTED EDGE LOADS SECTION
C
      READ(5,932) NEDGE
  932 FORMAT(I5)
      WRITE(6,912) NEDGE
  912 FORMAT(1H0,5X,21HNO. OF LOADED EDGES =,I5)
      WRITE(6,915)
  915 FORMAT(1H0,5X,38HLIST OF LOADED EDGES AND APPLIED LOADS)
      NODEG=3
      NCODE=NNODE
      IF(NNODE.EQ.4) NODEG=2
      IF(NNODE.EQ.9) NCODE=8
C
C*** LOOP OVER EACH LOADED EDGE
C
      DO 160 IEDGE=1,NEDGE
C
C*** READ DATA LOCATING THE LOADED EDGE AND APPLIED LOAD
C
      READ(5,902) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  902 FORMAT(4I5)
      WRITE(6,913) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  913 FORMAT(I10,5X,3I5)
      READ(5,914)  ((PRESS(IODEG,IDOFN),IDOFN=1,2),IODEG=1,NODEG)
      WRITE(6,914) ((PRESS(IODEG,IDOFN),IDOFN=1,2),IODEG=1,NODEG)
  914 FORMAT(6F10.3)
      ETASP=-1.0
C
C*** CALCULATE THE COORDINATES OF THE NODES OF THE ELEMENT EDGE
C
      DO 100 IODEG=1,NODEG
      LNODE=NOPRS(IODEG)
      DO 100 IDIME=1,2
  100 ELCOD(IDIME,IODEG)=COORD(LNODE,IDIME)
C
C*** ENTER LOOP FOR LINEAR NUMERICAL INTEGRATION
      DO 150 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS
C
```

```
      CALL      SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
C
C*** CALCULATE COMPONENTS OF THE EQUIVALENT NODAL LOADS
C
      DO 110 IDOFN=1,2
      PGASH(IDOFN)=0.0
      DGASH(IDOFN)=0.0
      DO 110 IODEG=1,NODEG
      PGASH(IDOFN)=PGASH(IDOFN)+PRESS(IODEG,IDOFN)*SHAPE(IODEG)
  110 DGASH(IDOFN)=DGASH(IDOFN)+ELCOD(IDOFN,IODEG)*DERIV(1,IODEG)
      DVOLU=WEIGP(IGAUS)
      PXCOM=DGASH(1)*PGASH(2)-DGASH(2)*PGASH(1)
      PYCOM=DGASH(1)*PGASH(1)+DGASH(2)*PGASH(2)
      IF(NTYPE.NE.3) GO TO 115
      RADUS=0.0
      DO 125 IODEG=1,NODEG
  125 RADUS=RADUS+SHAPE(IODEG)*ELCOD(1,IODEG)
      DVOLU=DVOLU*TWOPI*RADUS
  115 CONTINUE
C
C*** ASSOCIATE THE EQUIVALENT NODAL EDGE LOADS WITH AN ELEMENT
C
      DO 120 INODE=1,NNODE
      NLOCA=IABS(LNODS(NEASS,INODE))
  120 IF(NLOCA.EQ.NOPRS(1)) GO TO 130
  130 JNODE=INODE+NODEG-1
      KOUNT=0
      DO 140 KNODE=INODE,JNODE
      KOUNT=KOUNT+1
      NGASH=(KNODE-1)*NDOFN+1
      MGASH=(KNODE-1)*NDOFN+2
      IF(KNODE.GT.NCODE) NGASH=1
      IF(KNODE.GT.NCODE) MGASH=2
      RLOAD(NEASS,NGASH)=RLOAD(NEASS,NGASH)+SHAPE(KOUNT)*PXCOM*DVOLU
  140 RLOAD(NEASS,MGASH)=RLOAD(NEASS,MGASH)+SHAPE(KOUNT)*PYCOM*DVOLU
  150 CONTINUE
  160 CONTINUE
  700 CONTINUE
      WRITE(6,907)
  907 FORMAT(1H0,5X,36H TOTAL NODAL FORCES FOR EACH ELEMENT)
      DO 290 IELEM=1,NELEM
  290 WRITE(6,905) IELEM,(RLOAD(IELEM,IEVAB),IEVAB=1,NEVAB)
  905 FORMAT(1X,I4,5X,8E12.4/(10X,8E12.4))
      RETURN
      END
```

6.4.6 Subroutine LOADPB for evaluating the element nodal forces for plate bending applications

For plate bending applications two forms of loading will be considered. Firstly load components corresponding to the permissible generalised forces may be prescribed at the nodal points. Thus with respect to Fig. 6.9, a load in the z direction and couples acting in both the xz and yz planes may be input at each nodal point. Secondly a uniformly distributed load acting normal to the plate (i.e. in the z direction) may be applied. As in Section 6.4.5 such a loading must be converted into equivalent nodal forces before equation solution takes place. The equivalent nodal forces for node i take the form[4]

$$\begin{bmatrix} P_i \\ M_{xi} \\ M_{yi} \end{bmatrix}^{(e)} = \int_{A^{(e)}} N_i^{(e)} \begin{bmatrix} q \\ 0 \\ 0 \end{bmatrix} dA, \tag{6.64}$$

where q is the distributed load intensity and integration is taken over the element area. The structure of the subroutine is similar to that of subroutine LOADPS described in Section 6.4.5.

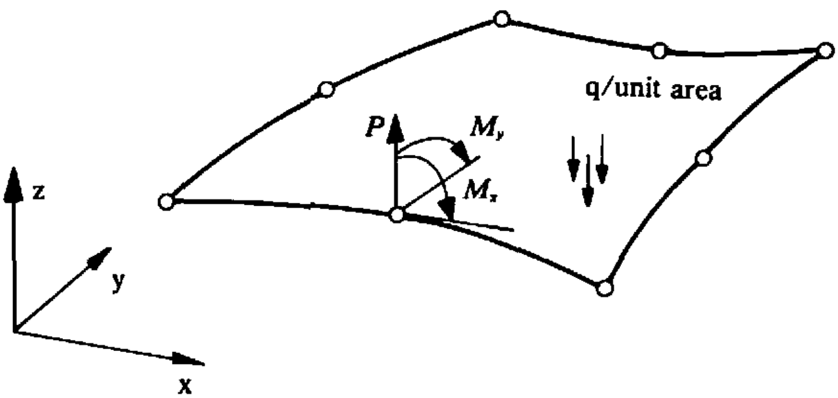


Fig. 6.9 Applied nodal and distributed forces for plate applications

```
      SUBROUTINE LOADPB    (COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,
     .                       NELEM,NEVAB,NGAUS,NNODE,NPOIN,PROPS,
     .                       RLOAD)
C***********************************************************************
C
C***  COMPUTE NODAL FORCES AFTER READING RELEVANT DATA
C***  FOR MINDLIN PLATE ELEMENTS
C
C***********************************************************************
      DIMENSION CARTD(2,9),COORD(MPOIN,2),DERIV(2,9),ELCOD(2,9),
     .          GPCOD(2,9),LNODS(MELEM,9),MATNO(MELEM),
     .          POINT(3),POSGP(4),PROPS(MMATS,8),RLOAD(MELEM,27),
     .          SHAPE(9),TITLE(12),WEIGP(4)
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 RLOAD(IELEM,IEVAB)=0.0
      READ(5,901) TITLE
  901 FORMAT(12A6)
      WRITE(6,903) TITLE
  903 FORMAT(1H0,12A6)
C
C*** READ DATA CONTROLLING LOADING TYPES TO BE INPUTTED
C
        READ(5,919)IPLOD
       WRITE(6,919)IPLOD
  919 FORMAT(4I5)
C
C*** READ NODAL POINT LOADS
C
      IF(IPLOD.EQ.0) GO TO 500
   20 READ(5,931) LODPT,(POINT(IDOFN),IDOFN=1,3)
      WRITE(6,931) LODPT,(POINT(IDOFN),IDOFN=1,3)
  931 FORMAT(I5,2F10.3)
C
```

```
C*** ASSOCIATE THE NODAL POINT LOADS WITH AN ELEMENT
C
      DO 30 IELEM=1,NELEM
      DO 30 INODE=1,NNODE
      NLOCA=IABS(LNODS(IELEM,INODE))
   30 IF(LODPT.EQ.NLOCA) GO TO 40
   40 DO 50 IDOFN=1,3
      NGASH=(INODE-1)*3+IDOFN
   50 RLOAD(IELEM,NGASH)=POINT(IDOFN)
      IF(LODPT.LT.NPOIN) GO TO 20
  500 CONTINUE
C
C***    LOOP OVER EACH ELEMENT
C
       DO 220  IELEM=1,NELEM
       LPROP=MATNO(IELEM)
       UDLOD=PROPS(LPROP,4)
       IF(UDLOD.EQ.0.0)GO TO 220
C
C***  EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 140 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      LNODE=IABS(LNODE)
      DO 140 IDIME=1,2
      ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
  140 CONTINUE
      KGASP=0
      CALL GAUSSQ     (NGAUS,POSGP,WEIGP)
C
C***  ENTER  LOOPS  FOR NUMERICAL  INTEGRATION
C
      DO 200 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      DO 200 JGAUS=1,NGAUS
      ETASP=POSGP(JGAUS)
      KGASP=KGASP+1
C
C***  EVALUATE THE SHAPE FUNCTIONS AT THE SAMPLING
C           POINTS AND ELEMENTAL AREA
C
      CALL        SFR2       (DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL        JACOB2     (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                        KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
C
C***   CALCULATE LOADS  AND ASSOCIATE WITH ELEMENT NODALPOINTS
C
      DO 180 INODE=1,NNODE
      NPOSN=(INODE-1)*3+1
      RLOAD(IELEM,NPOSN)=RLOAD(IELEM,NPOSN)+
     . SHAPE(INODE)*UDLOD*DAREA
  180 CONTINUE
  200 CONTINUE
  220 CONTINUE
      WRITE(6,907)
  907 FORMAT(1H0,5X,36H TOTAL NODAL FORCES FOR EACH ELEMENT)
      DO 290 IELEM=1,NELEM
  290 WRITE(6,905) IELEM,(RLOAD(IELEM,IEVAB),IEVAB=1,NEVAB)
  905 FORMAT(1X,I4,5X,8E12.4/(10X,8E12.4))
      RETURN
      END
```

6.4.7 Subroutine BMATPS for evaluating the strain matrix *B* for plane and axisymmetric situations

The function of this subroutine is to evaluate the strain matrix *B* at any position within an element. The relevant expressions are given in Table 6.1. The *B* matrix is stored in array BMATX ( ).

```
      SUBROUTINE BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE STRAIN-DISPLACEMENT MATRIX
C
C***********************************************************************
      DIMENSION BMATX(4,18),CARTD(2,9),SHAPE(9),GPCOD(2,9)
      NGASH=0
      DO 10 INODE=1,NNODE
      MGASH=NGASH+1
      NGASH=MGASH+1
      BMATX(1,MGASH)=CARTD(1,INODE)
      BMATX(1,NGASH)=0.0
      BMATX(2,MGASH)=0.0
      BMATX(2,NGASH)=CARTD(2,INODE)
      BMATX(3,MGASH)=CARTD(2,INODE)
      BMATX(3,NGASH)=CARTD(1,INODE)
      IF(NTYPE.NE.3) GO TO 10
      BMATX(4,MGASH)=SHAPE(INODE)/GPCOD(1,KGASP)
      BMATX(4,NGASH)=0.0
   10 CONTINUE
      RETURN
      END
```

6.4.8 Subroutine BMATPB for evaluating the strain matrix *B* for plate bending problems

This subroutine evaluates the strain matrix *B* within any point of an element for plate bending applications according to Table 6.1. The *B* matrix is partitioned into plane, BPLAN, flexural, BFLEX, and shear, BSHER, contributions.

```
      SUBROUTINE BMATPB    (BFLEX,BPLAN,BSHER,CARTD,KNODE,SHAPE,
     .                      IFPLA,IFFLE,IFSHE)
C***********************************************************************
C
C*** EVALUATES STRAIN-DISPLACEMENT MATRIX FOR
C*** MINDLIN PLATE
C
C***********************************************************************
      DIMENSION  BFLEX(3,3),BPLAN(3,2),BSHER(2,3),
     .           CARTD(2,9),SHAPE(9)
      DNKDX=CARTD(1,KNODE)
      DNKDY=CARTD(2,KNODE)
C*** FORM BPLAN
      IF(IFPLA.EQ.0) GO TO 10
      DO 1 IROWS=1,3
      DO 1 JCOLS=1,2
    1 BPLAN(IROWS,JCOLS)=0.0
      BPLAN(1,1)=DNKDX
      BPLAN(2,2)=DNKDY
      BPLAN(3,1)=DNKDY
      BPLAN(3,2)=DNKDX
```

```
C*** FORM BFLEX
   10 IF(IFFLE.EQ.0) GO TO 20
      DO 2 IROWS=1,3
      DO 2 JCOLS=1,3
    2 BFLEX(IROWS,JCOLS)=0.0
      BFLEX(1,2)=-DNKDX
      BFLEX(2,3)=-DNKDY
      BFLEX(3,2)=-DNKDY
      BFLEX(3,3)=-DNKDX
C*** FORM BSHER
   20 IF(IFSHE.EQ.0) RETURN
      DO 3 IROWS=1,2
      DO 3 JCOLS=1,3
    3 BSHER(IROWS,JCOLS)=0.0
      BSHER(1,1)=DNKDX
      BSHER(1,2)=-SHAPE(KNODE)
      BSHER(2,1)=DNKDY
      BSHER(2,3)=-SHAPE(KNODE)
      RETURN
      END
```

6.4.9 Subroutine MODPS for evaluating the *D* matrix for plane and axisymmetric situations

This subroutine simply evaluates the elasticity matrix ***D*** for either plane stress, plane strain or axisymmetric situations according to (6.7), (6.16) or (6.24) respectively. The ***D*** matrix is stored in the array DMATX ( ).

```
      SUBROUTINE MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE D-MATRIX
C
C***********************************************************************
      DIMENSION DMATX(4,4),PROPS(MMATS,7)
      YOUNG=PROPS(LPROP,1)
      POISS=PROPS(LPROP,2)
      DO 10 ISTR1=1,4
      DO 10 JSTR1=1,4
   10 DMATX(ISTR1,JSTR1)=0.0
      IF(NTYPE.NE.1) GO TO 4
C
C***  D MATRIX FOR PLANE STRESS CASE
C
      CONST=YOUNG/(1.0-POISS*POISS)
      DMATX(1,1)=CONST
      DMATX(2,2)=CONST
      DMATX(1,2)=CONST*POISS
      DMATX(2,1)=CONST*POISS
      DMATX(3,3)=(1.0-POISS)*CONST/2.0
      RETURN
    4 IF(NTYPE.NE.2) GO TO 6
C
C***  D MATRIX FOR PLANE STRAIN CASE
C
      CONST=YOUNG*(1.0-POISS)/((1.0+POISS)*(1.0-2.0*POISS))
      DMATX(1,1)=CONST
      DMATX(2,2)=CONST
      DMATX(1,2)=CONST*POISS/(1.0-POISS)
      DMATX(2,1)=CONST*POISS/(1.0-POISS)
```

```
      DMATX(3,3)=(1.0-2.0*POISS)*CONST/(2.0*(1.0-POISS))
      RETURN
    6 IF(NTYPE.NE.3) GO TO 8
C
C***  D MATRIX FOR AXISYMMETRIC CASE
C
      CONST=YOUNG*(1.0-POISS)/((1.0+POISS)*(1.0-2.0*POISS))
      CONSS=POISS/(1.0-POISS)
      DMATX(1,1)=CONST
      DMATX(2,2)=CONST
      DMATX(3,3)=CONST*(1.0-2.0*POISS)/(2.0*(1.0-POISS))
      DMATX(1,2)=CONST*CONSS
      DMATX(1,4)=CONST*CONSS
      DMATX(2,1)=CONST*CONSS
      DMATX(2,4)=CONST*CONSS
      DMATX(4,1)=CONST*CONSS
      DMATX(4,2)=CONST*CONSS
      DMATX(4,4)=CONST
    8 CONTINUE
      RETURN
      END
```

6.4.10 Subroutine MODPB for evaluating the *D* matrix for plate bending applications

This subroutine evaluates the elasticity matrix ***D*** for plate bending situations according to (6.35). Again the result is partitioned into plane, DPLAN, flexural, DFLEX, and shear, DSHER, contributions.

```
      SUBROUTINE MODPB     (DFLEX,DPLAN,DSHER,LPROP,MMATS,PROPS,
     .                       IFPLA,IFFLE,IFSHE)
C**********************************************************************
C
C*** CALCULATES MATRIX OF ELASTIC RIGIDITIES
C*** FOR MINDLIN PLATE
C
C**********************************************************************
      DIMENSION DFLEX(3,3),DPLAN(3,3),DSHER(2,2),
     .          PROPS(MMATS,8)
      YOUNG=PROPS(LPROP,1)
      POISS=PROPS(LPROP,2)
      THICK=PROPS(LPROP,3)
C*** FORM DPLAN
      IF(IFPLA.EQ.0) GO TO 10
      DO 1 IROWS=1,3
      DO 1 JCOLS=1,3
    1 DPLAN(IROWS,JCOLS)=0.0
      CONST=(YOUNG*THICK)/(1.0-POISS*POISS)
      DPLAN(1,1)=CONST
      DPLAN(2,2)=CONST
      DPLAN(1,2)=CONST*POISS
      DPLAN(2,1)=CONST*POISS
      DPLAN(3,3)=CONST*(1.0-POISS)/2.0
C*** FORM DFLEX
   10 IF(IFFLE.EQ.0) GOTO 20
      DO 2 IROWS=1,3
      DO 2 JCOLS=1,3
    2 DFLEX(IROWS,JCOLS)=0.0
      CONST=(YOUNG*THICK**3)/(12.*(1.-POISS*POISS))
      DFLEX(1,1)=CONST
      DFLEX(2,2)=CONST
      DFLEX(1,2)=CONST*POISS
```

```
      DFLEX(2,1)=CONST*POISS
      DFLEX(3,3)=CONST*(1.-POISS)/2.
C*** FORM DSHER
   20 IF(IFSHE.EQ.0) RETURN
      DO 3 IROWS=1,2
      DO 3 JCOLS=1,2
    3 DSHER(IROWS,JCOLS)=0.0
      DSHER(1,1)=(YOUNG*THICK)/(2.4+2.4*POISS)
      DSHER(2,2)=(YOUNG*THICK)/(2.4+2.4*POISS)
      RETURN
      END
```

6.4.11 Subroutine DBE for formulating the matrix product *DB*

This subroutine simply multiplies the elasticity matrix ***D*** by the strain matrix ***B***.

```
      SUBROUTINE DBE(BMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTRE,NSTR1)
C*********************************************************************
C
C**** THIS SUBROUTINE MULTIPLIES THE D-MATRIX BY THE B-MATRIX
C
C*********************************************************************
      DIMENSION BMATX(NSTR1,MEVAB),DBMAT(NSTR1,MEVAB),
     .          DMATX(NSTR1,NSTR1)
      DO 2 ISTRE=1,NSTRE
      DO 2 IEVAB=1,NEVAB
      DBMAT(ISTRE,IEVAB)=0.0
      DO 2 JSTRE=1,NSTRE
      DBMAT(ISTRE,IEVAB)=DBMAT(ISTRE,IEVAB)+
     .DMATX(ISTRE,JSTRE)*BMATX(JSTRE,IEVAB)
    2 CONTINUE
      RETURN
      END
```

6.4.12 Subroutine FRONT for equation solution by the frontal method

The function of this subroutine is to assemble the contributions from each element to form the global stiffness matrix and global load vector and to solve the resulting set of simultaneous equations by Gaussian direct elimination. The main feature of the frontal solution technique is that it *assembles the equations and eliminates the variables at the same time*. Complete details of the frontal process can be found in Chapter 8, Ref. 4. The subroutine presented in Ref. 4 differs from the one listed in this section in three important ways:

- As described in Sections 3.3 and 3.4 for one-dimensional problems, a full equation solution need only be undertaken for iterations during which the element stiffnesses are being modified. Such a situation is recognised by the resolution counter KRESL = 1. On the other hand if the element stiffnesses have not been changed during the iteration, signified by KRESL = 2, only the R.H.S. or load terms need be reduced during the elimination phase. This situation is identical to the case of solution for second and subsequent loading cases in elastic problems.

● The reduced equations corresponding to eliminated variables are stored in core in a temporary array termed a *buffer area.* As soon as this array is full, the information is then transferred to disc. The number of reduced equations that can be accommodated in the buffer area is governed by the specified parameter, MBUFA. Thus on elimination of a variable a counter over the number of eliminated variables is incremented by one and the reduced equations stored in core. The counter is checked against the permissible buffer length, MBUFA. If this has been reached, the buffer array is transferred to disc file and the counter reset to zero. On back-substitution the contents of a complete buffer length are read from discfile by backspacing.

● The displacement and reaction values evaluated by subroutine FRONT during each iteration are incremental values and must be accumulated to give the total displacements, TDISP ( ) and total reactions, TREAC ( ). Also the incremental reactions must be added into the vector of total applied loads, TLOAD ( ), in order to check for convergence of the iteration process; since equilibrium is satisfied when the applied loads and reactions at restrained nodes balance with the nodal forces equivalent to the internal stress field.

The displacements and reactions evaluated in Subroutine FRONT are stored for output by Subroutine OUTPUT described in Section 7.8.8.

```
      SUBROUTINE FRONT(ASDIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,
     .                 IITER,GLOAD,GSTIF,LOCEL,LNODS,KRESL,MBUFA,MELEM,
     .                 MEVAB,MFRON,MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,
     .                 NDOFN,NELEM,NEVAB,NNODE,NOFIX,NPIVO,NPOIN,
     .                 NTOTV,TDISP,TLOAD,TREAC,VECRV)
C*********************************************************************
C
C**** THIS SUBROUTINE UNDERTAKES EQUATION SOLUTION BY THE FRONTAL
C     METHOD
C
C*********************************************************************
      DIMENSION ASDIS(MTOTV),ELOAD(MELEM,MEVAB),EQRHS(MBUFA),
     .          EQUAT(MFRON,MBUFA),ESTIF(MEVAB,MEVAB),FIXED(MTOTV),
     .          IFFIX(MTOTV),NPIVO(MBUFA),VECRV(MFRON),GLOAD(MFRON),
     .          GSTIF(MSTIF),LNODS(MELEM,9) ,LOCEL(MEVAB),NACVA(MFRON),
     .          NAMEV(MBUFA),NDEST(MEVAB),NOFIX(MVFIX),NOUTP(2),
     .          TDISP(MTOTV),TLOAD(MELEM,MEVAB),TREAC(MVFIX,NDOFN)
      NFUNC(I,J)=(J*J-J)/2+I
C
C*** CHANGE THE SIGN OF THE LAST APPEARANCE OF EACH NODE
C
      IF(IINCS.GT.1.OR.IITER.GT.1) GO TO 455
      DO 140 IPOIN=1,NPOIN
      KLAST=0
      DO 130 IELEM=1,NELEM
      DO 120 INODE=1,NNODE
      IF(LNODS(IELEM,INODE).NE.IPOIN) GO TO 120
      KLAST=IELEM
      NLAST=INODE
  120 CONTINUE
```

```
  130 CONTINUE
      IF(KLAST.NE.0) LNODS(KLAST,NLAST)=-IPOIN
  140 CONTINUE
  455 CONTINUE
C
C*** START BY INITIALIZING EVERYTHING THAT MATTERS TO ZERO
C
      DO 450 IBUFA=1,MBUFA
  450 EQRHS(IBUFA)=0.0
      DO 150 ISTIF=1,MSTIF
  150 GSTIF(ISTIF)=0.0
      DO 160 IFRON=1,MFRON
      GLOAD(IFRON)=0.0
      VECRV(IFRON)=0.0
      NACVA(IFRON)=0
      DO 160 IBUFA=1,MBUFA
  160 EQUAT(IFRON,IBUFA)=0.0
C
C*** AND PREPARE FOR DISC READING AND WRITING OPERATIONS
C
      NBUFA=0
      IF(KRESL.GT.1) NBUFA=MBUFA
      REWIND 1
      REWIND 2
      REWIND 3
      REWIND 4
      REWIND 8
C
C*** ENTER MAIN ELEMENT ASSEMBLY-REDUCTION LOOP
C
      NFRON=0
      KELVA=0
      DO 320 IELEM=1,NELEM
      IF(KRESL.GT.1) GO TO 400
      KEVAB=0
      READ(1) ESTIF
      DO 170 INODE=1,NNODE
      DO 170 IDOFN=1,NDOFN
      NPOSI=(INODE-1)*NDOFN+IDOFN
      LOCNO=LNODS(IELEM,INODE)
      IF(LOCNO.GT.0) LOCEL(NPOSI)=(LOCNO-1)*NDOFN+IDOFN
      IF(LOCNO.LT.0) LOCEL(NPOSI)=(LOCNO+1)*NDOFN-IDOFN
  170 CONTINUE
C
C*** START BY LOOKING FOR EXISTING DESTINATIONS
C
      DO 210 IEVAB=1,NEVAB
      NIKNO=IABS(LOCEL(IEVAB))
      KEXIS=0
      DO 180 IFRON=1,NFRON
      IF(NIKNO.NE.NACVA(IFRON)) GO TO 180
      KEVAB=KEVAB+1
      KEXIS=1
      NDEST(KEVAB)=IFRON
  180 CONTINUE
      IF(KEXIS.NE.0) GO TO 210
C
C*** WE NOW SEEK NEW EMPTY PLACES FOR DESTINATION VECTOR
C
      DO 190 IFRON=1,MFRON
      IF(NACVA(IFRON).NE.0) GO TO 190
      NACVA(IFRON)=NIKNO
      KEVAB=KEVAB+1
      NDEST(KEVAB)=IFRON
      GO TO 200
```

```
  190 CONTINUE
C
C*** THE NEW PLACES MAY DEMAND AN INCREASE IN CURRENT FRONTWIDTH
C
  200 IF(NDEST(KEVAB).GT.NFRON) NFRON=NDEST(KEVAB)
  210 CONTINUE
      WRITE(8) LOCEL,NDEST,NACVA,NFRON
  400 IF(KRESL.GT.1) READ(8) LOCEL,NDEST,NACVA,NFRON
C
C*** ASSEMBLE ELEMENT LOADS
C
      DO 220 IEVAB=1,NEVAB
      IDEST=NDEST(IEVAB)
      GLOAD(IDEST)=GLOAD(IDEST)+ELOAD(IELEM,IEVAB)
C
C*** ASSEMBLE THE ELEMENT STIFFNESSES-BUT NOT IN RESOLUTION
C
      IF(KRESL.GT.1) GO TO 402
      DO 222 JEVAB=1,IEVAB
      JDEST=NDEST(JEVAB)
      NGASH=NFUNC(IDEST,JDEST)
      NGISH=NFUNC(JDEST,IDEST)
      IF(JDEST.GE.IDEST) GSTIF(NGASH)=GSTIF(NGASH)+ESTIF(IEVAB,JEVAB)
      IF(JDEST.LT.IDEST) GSTIF(NGISH)=GSTIF(NGISH)+ESTIF(IEVAB,JEVAB)
  222 CONTINUE
  402 CONTINUE
  220 CONTINUE
C
C*** RE-EXAMINE EACH ELEMENT NODE, TO ENQUIRE WHICH CAN BE ELIMINATED
C
      DO 310 IEVAB=1,NEVAB
      NIKNO=-LOCEL(IEVAB)
      IF(NIKNO.LE.0) GO TO 310
C
C*** FIND POSITIONS OF VARIABLES READY FOR ELIMINATION
C
      DO 300 IFRON=1,NFRON
      IF(NACVA(IFRON).NE.NIKNO) GO TO 300
      NBUFA=NBUFA+1
C
C*** WRITE EQUATIONS TO DISC OR TO TAPE
C
      IF(NBUFA.LE.MBUFA) GO TO 406
      NBUFA=1
      IF(KRESL.GT.1) GO TO 408
      WRITE(2) EQUAT,EQRHS,NPIVO,NAMEV
      GO TO 406
  408 WRITE(4) EQRHS
      READ(2) EQUAT,EQRHS,NPIVO,NAMEV
  406 CONTINUE
C
C*** EXTRACT THE COEFFICIENTS OF THE NEW EQUATION FOR ELIMINATION
C
      IF(KRESL.GT.1) GO TO 404
      DO 230 JFRON=1,MFRON
      IF(IFRON.LT.JFRON) NLOCA=NFUNC(IFRON,JFRON)
      IF(IFRON.GE.JFRON) NLOCA=NFUNC(JFRON,IFRON)
      EQUAT(JFRON,NBUFA)=GSTIF(NLOCA)
  230 GSTIF(NLOCA)=0.0
  404 CONTINUE
C
C*** AND EXTRACT THE CORRESPONDING RIGHT HAND SIDES
C
      EQRHS(NBUFA)=GLOAD(IFRON)
      GLOAD(IFRON)=0.0
```

```
      KELVA=KELVA+1
      NAMEV(NBUFA)=NIKNO
      NPIVO(NBUFA)=IFRON
C
C*** DEAL WITH PIVOT
C
      PIVOT=EQUAT(IFRON,NBUFA)
      IF(PIVOT.GT.0.0) GO TO 235
      WRITE(6,900) NIKNO,PIVOT
  900 FORMAT(1H0,3X,52HNEGATIVE OR ZERO PIVOT ENCOUNTERED FOR VARIABLE N
     .O. ,I4,10H OF VALUE ,E17.6)
      STOP
  235 CONTINUE
      EQUAT(IFRON,NBUFA)=0.0
C
C*** ENQUIRE WHETHER PRESENT VARIABLE IS FREE OR PRESCRIBED
C
      IF(IFFIX(NIKNO).EQ.0) GO TO 250
C
C*** DEAL WITH A PRESCRIBED DEFLECTION
C
      DO 240 JFRON=1,NFRON
  240 GLOAD(JFRON)=GLOAD(JFRON)-FIXED(NIKNO)*EQUAT(JFRON,NBUFA)
      GO TO 280
C
C*** ELIMINATE A FREE VARIABLE - DEAL WITH THE RIGHT HAND SIDE FIRST
C
  250 DO 270 JFRON=1,NFRON
      GLOAD(JFRON)=GLOAD(JFRON)-EQUAT(JFRON,NBUFA)*EQRHS(NBUFA)/PIVOT
C
C*** NOW DEAL WITH THE COEFFICIENTS IN CORE
C
      IF(KRESL.GT.1) GO TO 418
      IF(EQUAT(JFRON,NBUFA).EQ.0.0) GO TO 270
      NLOCA=NFUNC(0,JFRON)
      CUREQ=EQUAT(JFRON,NBUFA)
      DO 260 LFRON=1,JFRON
      NGASH=LFRON+NLOCA
  260 GSTIF(NGASH)=GSTIF(NGASH)-CUREQ*EQUAT(LFRON,NBUFA)
     . /PIVOT
  418 CONTINUE
  270 CONTINUE
  280 EQUAT(IFRON,NBUFA)=PIVOT
C
C*** RECORD THE NEW VACANT SPACE, AND REDUCE FRONTWIDTH IF POSSIBLE
C
      NACVA(IFRON)=0
      GO TO 290
C
C*** COMPLETE THE ELEMENT LOOP IN THE FORWARD ELIMINATION
C
  300 CONTINUE
  290 IF(NACVA(NFRON).NE.0) GO TO 310
      NFRON=NFRON-1
      IF(NFRON.GT.0) GO TO 290
  310 CONTINUE
  320 CONTINUE
      IF(KRESL.EQ.1) WRITE(2) EQUAT,EQRHS,NPIVO,NAMEV
      BACKSPACE 2
C
C*** ENTER BACK-SUBSTITUTION PHASE. LOOP BACKWARDS THROUGH VARIABLES
C
      DO 340 IELVA=1,KELVA
C
C***READ A NEW BLOCK OF EQUATIONS - IF NEEDED
```

```
C
      IF(NBUFA.NE.0) GO TO 412
      BACKSPACE 2
      READ(2) EQUAT,EQRHS,NPIVO,NAMEV
      BACKSPACE 2
      NBUFA=MBUFA
      IF(KRESL.EQ.1) GO TO 412
      BACKSPACE 4
      READ(4) EQRHS
      BACKSPACE 4
  412 CONTINUE
C
C*** PREPARE TO BACK-SUBSTITUTE FROM THE CURRENT EQUATION
C
      IFRON=NPIVO(NBUFA)
      NIKNO=NAMEV(NBUFA)
      PIVOT=EQUAT(IFRON,NBUFA)
      IF(IFFIX(NIKNO).NE.0) VECRV(IFRON)=FIXED(NIKNO)
      IF(IFFIX(NIKNO).EQ.0) EQUAT(IFRON,NBUFA)=0.0
C
C*** BACK-SUBSTITUTE IN THE CURRENT EQUATION
C
      DO 330 JFRON=1,MFRON
  330 EQRHS(NBUFA)=EQRHS(NBUFA)-VECRV(JFRON)*EQUAT(JFRON,NBUFA)
C
C*** PUT THE FINAL VALUES WHERE THEY BELONG
C
      IF(IFFIX(NIKNO).EQ.0) VECRV(IFRON)=EQRHS(NBUFA)/PIVOT
      IF(IFFIX(NIKNO).NE.0) FIXED(NIKNO)=-EQRHS(NBUFA)
      NBUFA=NBUFA-1
      ASDIS(NIKNO)=VECRV(IFRON)
  340 CONTINUE
C
C*** ADD DISPLACEMENTS TO PREVIOUS TOTAL VALUES
C
      DO 345 ITOTV=1,NTOTV
  345 TDISP(ITOTV)=TDISP(ITOTV)+ASDIS(ITOTV)
C
C*** STORE REACTIONS FOR PRINTING LATER
C
      KBOUN=1
      DO 370 IPOIN=1,NPOIN
      NLOCA=(IPOIN-1)*NDOFN
      DO 350 IDOFN=1,NDOFN
      NGUSH=NLOCA+IDOFN
      IF(IFFIX(NGUSH).GT.0) GO TO 360
  350 CONTINUE
      GO TO 370
  360 DO 510 IDOFN=1,NDOFN
      NGASH=NLOCA+IDOFN
  510 TREAC(KBOUN,IDOFN)=TREAC(KBOUN,IDOFN)+FIXED(NGASH)
      KBOUN=KBOUN+1
  370 CONTINUE
C
C*** ADD REACTIONS INTO THE TOTAL LOAD ARRAY
C
      DO 700 IPOIN=1,NPOIN
      DO 710 IELEM=1,NELEM
      DO 710 INODE=1,NNODE
      NLOCA=IABS(LNODS(IELEM,INODE))
  710 IF(IPOIN.EQ.NLOCA) GO TO 720
  720 DO 730 IDOFN=1,NDOFN
      NGASH=(INODE-1)*NDOFN+IDOFN
      MGASH=(IPOIN-1)*NDOFN+IDOFN
  730 TLOAD(IELEM,NGASH)=TLOAD(IELEM,NGASH)+FIXED(MGASH)
```

```
  700 CONTINUE
      RETURN
      END
```

6.4.13 Data error diagnostic subroutine CHECK1

The function of this subroutine is to scrutinise the problem control parameters, which are accepted by the data input subroutine, INPUT, which will be described in Section 6.5.1. Since subroutine INPUT is common to plane stress/strain, axisymmetric and plate bending applications, subroutine CHECK1 will only check that the control parameters are within the bounds defined by the correct values for the four cases.

A counter, KEROR, is employed to indicate whether or not any errors have been detected. If errors have been found (indicated by KEROR = 1), subroutine ECHO, described in the next section, is called to list the remainder of the input data.

Any errors detected are signalled by means of printed error numbers. The interpretation of each error message is given in Table 6.2.

```
      SUBROUTINE CHECK1(NDOFN,NELEM,NGAUS,NMATS,NNODE,NPOIN,
     .                  NSTRE,NTYPE,NVFIX,NCRIT,NALGO,NINCS)
C***********************************************************************
C
C**** THIS SUBROUTINE CHECKS THE MAIN CONTROL DATA
C
C***********************************************************************
      DIMENSION NEROR(24)
      DO 10 IEROR=1,12
   10 NEROR(IEROR)=0
C
C*** CREATE THE DIAGNOSTIC MESSAGES
C
      IF(NPOIN.LE.0) NEROR(1)=1
      IF(NELEM*NNODE.LT.NPOIN) NEROR(2)=1
      IF(NVFIX.LT.2.OR.NVFIX.GT.NPOIN) NEROR(3)=1
      IF(NINCS.LT.1) NEROR(4)=1
      IF(NTYPE.LT.1.OR.NTYPE.GT.3) NEROR(5)=1
      IF(NNODE.LT.4.OR.NNODE.GT.9) NEROR(6)=1
      IF(NDOFN.LT.2.OR.NDOFN.GT.5) NEROR(7)=1
      IF(NMATS.LT.1.OR.NMATS.GT.NELEM) NEROR(8)=1
      IF(NCRIT.LT.1.OR.NCRIT.GT.4) NEROR(9)=1
      IF(NGAUS.LT.2.OR.NGAUS.GT.3) NEROR(10)=1
      IF(NALGO.LT.1.OR.NALGO.GT.4) NEROR(11)=1
      IF(NSTRE.LT.3.OR.NSTRE.GT.5) NEROR(12)=1
C
C*** EITHER RETURN,OR ELSE PRINT THE ERRORS DIAGNOSED
C
      KEROR=0
      DO 20 IEROR=1,12
      IF(NEROR(IEROR).EQ.0) GO TO 20
      KEROR=1
      WRITE(6,900) IEROR
  900 FORMAT(//31H *** DIAGNOSIS BY CHECK1, ERROR,I3)
   20 CONTINUE
      IF(KEROR.EQ.0) RETURN
```

```
C
C*** OTHERWISE ECHO ALL THE REMAINING DATA WITHOUT FURTHER COMMENT
C
      CALL ECHO
      END
```

Table 6.2 Errors diagnosed by Subroutine CHECK1.

Error Label	Interpretation
1	The specified total number of node points, NPOIN, in the structure is less than or equal to zero.
2	The possible maximum total number of node points in the structure is less than the specified total, NPOIN.
3	The number of restrained nodal points is less than 2 or greater than NPOIN (for plane problems at least 2 points must be restrained to eliminate rigid body motions).
4	The total number of load increments is less than 1.
5	The problem type parameter, NTYPE, is not specified as either 1, 2 or 3.
6	The number of nodes/element is less than 4 (linear quadrilateral) or greater than 9 (quadratic Lagrangian elements).
7	The number of degrees of freedom per node is not equal to 2 (plane) or 3 (plate problems).
8	The total number of different materials is less than or equal to zero or greater than the total number of elements in the structure.
9	The parameter specifying the yield criterion to be employed is outside the permissible range.
10	The number of Gaussian integration points in each direction is not equal to either 2 or 3.
11	The parameter specifying the nonlinear solution algorithm to be employed is outside the permissible range.
12	The size of the stress matrix is less than 3 (plane) or greater than 5 (plate problems).

6.4.14 Data echo subroutine, ECHO

The function of this subroutine is to list all the remaining data cards after at least one error has been detected by either of the diagnostic subroutines CHECK1 or CHECK2. This is accomplished by means of a simple read and write operation in alphanumeric format.

```
      SUBROUTINE ECHO
C***********************************************************************
C
C**** IF DATA ERRORS HAVE BEEN DETECTED BY SUBROUTINES CHECK1 OR
C     CHECK2,THIS SUBROUTINE READS AND WRITES THE REMAINING DATA CARDS
C
C***********************************************************************
      DIMENSION NTITL(80)
      WRITE(6,900)
```

```
900 FORMAT(//50H NOW FOLLOWS A LISTING OF POST-DISASTER DATA CARDS/)
 10 READ(5,905) NTITL
905 FORMAT(80A1)
    WRITE(6,910) NTITL
910 FORMAT(20X,80A1)
    GO TO 10
    END
```

6.4.15 Data error diagnostic subroutine, CHECK2

If the problem control parameters have passed the scrutiny of subroutine CHECK1, the geometric data, boundary conditions and material properties are then assimilated by subroutine INPUT. This data is then scrutinised for possible errors in subroutine CHECK2 where error types 13 to 24, listed in Table 6.3, are checked for.

Probably the most useful check in this subroutine is the one which ensures that the maximum frontwidth does not exceed the dimensions specified in subroutine FRONT. Subroutine CHECK2 is described in detail in Chapter 9, Ref. 4.

```
      SUBROUTINE CHECK2(COORD,IFFIX,LNODS,MATNO,MELEM,MFRON,MPOIN,MTOTV,
     .                  MVFIX,NDFRO,NDOFN,NELEM,NMATS,NNODE,NOFIX,NPOIN,
     .                  NVFIX)
C*********************************************************************
C
C**** THIS SUBROUTINE CHECKS THE REMAINDER OF THE INPUT DATA
C
C*********************************************************************
      DIMENSION COORD(MPOIN,2),IFFIX(MTOTV),LNODS(MELEM,9),
     .          MATNO(MELEM),NDFRO(MELEM),NEROR(24),NOFIX(MVFIX)
C               IFPRE(NVFIX,NDOFN)
C*** CHECK AGAINST TWO IDENTICAL NONZERO NODAL COORDINATES
C
      DO 5 IEROR=13,24
    5 NEROR(IEROR)=0
      DO 10 IELEM=1,NELEM
   10 NDFRO(IELEM)=0
      DO 50 IPOIN=2,NPOIN
      KPOIN=IPOIN-1
      DO 30 JPOIN=1,KPOIN
      DO 20 IDIME=1,2
      IF(COORD(IPOIN,IDIME).NE.COORD(JPOIN,IDIME)) GO TO 30
   20 CONTINUE
      NEROR(13)=NEROR(13)+1
   30 CONTINUE
   40 CONTINUE
C
C*** CHECK THE LIST OF ELEMENT PROPERTY NUMBERS
C
      DO 50 IELEM=1,NELEM
   50 IF(MATNO(IELEM).LE.0.OR.MATNO(IELEM).GT.NMATS) NEROR(14)=NEROR(14)
     . +1
C
C*** CHECK FOR IMPOSSIBLE NODE NUMBERS
C
      DO 70 IELEM=1,NELEM
      DO 60 INODE=1,NNODE
      IF(LNODS(IELEM,INODE).EQ.0) NEROR(15)=NEROR(15)+1
```

```
   60 IF(LNODS(IELEM,INODE).LT.0.OR.LNODS(IELEM,INODE).GT.NPOIN) NEROR(
     . 16)=NEROR(16)+1
   70 CONTINUE
C
C*** CHECK FOR ANY REPETITION OF A NODE NUMBER WITHIN AN ELEMENT
C
      DO 140 IPOIN=1,NPOIN
      KSTAR=0
      DO 100 IELEM=1,NELEM
      KZERO=0
      DO 90 INODE=1,NNODE
      IF(LNODS(IELEM,INODE).NE.IPOIN) GO TO 90
      KZERO=KZERO+1
      IF(KZERO.GT.1) NEROR(17)=NEROR(17)+1
C
C*** SEEK FIRST,LAST AND INTERMEDIATE APPEARANCES OF NODE IPOIN
C
      IF(KSTAR.NE.0) GO TO 80
      KSTAR=IELEM
C
C*** CALCULATE INCREASE OR DECREASE IN FRONTWIDTH AT EACH ELEMENT STAGE
C
      NDFRO(IELEM)=NDFRO(IELEM)+NDOFN
   80 CONTINUE
C
C*** AND CHANGE THE SIGN OF THE LAST APPEARANCE OF EACH NODE
C
      KLAST=IELEM
      NLAST=INODE
   90 CONTINUE
  100 CONTINUE
      IF(KSTAR.EQ.0) GO TO 110
      IF(KLAST.LT.NELEM) NDFRO(KLAST+1)=NDFRO(KLAST+1)-NDOFN
      LNODS(KLAST,NLAST)=-IPOIN
      GO TO 140
C
C*** CHECK THAT COORDINATES FOR AN UNUSED NODE HAVE NOT BEEN SPECIFIED
C
  110 WRITE(6,900) IPOIN
  900 FORMAT(/15H CHECK WHY NODE,I4,14H NEVER APPEARS)
      NEROR(18)=NEROR(18)+1
      SIGMA=0.0
      DO 120 IDIME=1,2
  120 SIGMA=SIGMA+ABS(COORD(IPOIN,IDIME))
      IF(SIGMA.NE.0.0) NEROR(19)=NEROR(19)+1
C
C*** CHECK THAT AN UNUSED NODE NUMBER IS NOT A RESTRAINED NODE
C
      DO 130 IVFIX=1,NVFIX
  130 IF(NOFIX(IVFIX).EQ.IPOIN) NEROR(20)=NEROR(20)+1
  140 CONTINUE
C
C*** CALCULATE THE LARGEST FRONTWIDTH
C
      NFRON=0
      KFRON=0
      DO 150 IELEM=1,NELEM
      NFRON=NFRON+NDFRO(IELEM)
  150 IF(NFRON.GT.KFRON) KFRON=NFRON
      WRITE(6,905) KFRON
  905 FORMAT(//33H MAXIMUM FRONTWIDTH ENCOUNTERED =,I5)
      IF(KFRON.GT.MFRON) NEROR(21)=1
C
C*** CONTINUE CHECKING THE DATA FOR THE FIXED VALUES
C
```

```
      DO 170 IVFIX=1,NVFIX
      IF(NOFIX(IVFIX).LE.0.OR.NOFIX(IVFIX).GT.NPOIN) NEROR(22)=NEROR(22)
     . +1
      KOUNT=0
      NLOCA=(NOFIX(IVFIX)-1)*NDOFN
      DO 160 IDOFN=1,NDOFN
      NLOCA=NLOCA+1
  160 IF(IFFIX(NLOCA).GT.0) KOUNT=1
      IF(KOUNT.EQ.0) NEROR(23)=NEROR(23)+1
      KVFIX=IVFIX-1
      DO 170 JVFIX=1,KVFIX
  170 IF(IVFIX.NE.1.AND.NOFIX(IVFIX).EQ.NOFIX(JVFIX)) NEROR(24)=NEROR(24
     . )+1
      KEROR=0
      DO 180 IEROR=13,24
      IF(NEROR(IEROR).EQ.0) GO TO 180
      KEROR=1
      WRITE(6,910) IEROR,NEROR(IEROR)
  910 FORMAT(//31H *** DIAGNOSIS BY CHECK2, ERROR,I3,6X,18H ASSOCIATED N
     .UMBER,I5)
  180 CONTINUE
      IF(KEROR.NE.0) GO TO 200
C
C*** RETURN ALL NODAL CONNECTION NUMBERS TO POSITIVE VALUES
C
      DO 190 IELEM=1,NELEM
      DO 190 INODE=1,NNODE
  190 LNODS(IELEM,INODE)=IABS(LNODS(IELEM,INODE))
      RETURN
  200 CALL ECHO
      END
```

Table 6.3 Errors diagnosed by Subroutine CHECK2

Error Label	Interpretation
13	A total of x identical nodal coordinates have been detected, i.e. x nodal points have coordinates which are identical to those of one or more of the remaining nodes.
14	A total of x element material identification numbers are less than or equal to zero or greater than the total number of elements in the structure.
15	A total of x nodal connection numbers have a zero value.
16	A total of x nodal connection numbers are negative or greater than the specified maximum value, NPOIN.
17	A total of x repetitions of node numbers within individual elements have been detected.
18	A total of x nodes exist in the list of nodal points which do not appear anywhere in the list of element nodal connection numbers.
19	Non-zero coordinates have been specified for a total of x nodes which do not appear in the list of element nodal connection numbers.
20	A total of x node numbers which do not appear in the element nodal connections list have been specified as restrained nodal points.
21	The largest frontwidth encountered in the problem has exceeded the maximum value specified in subroutine FRONT of the program.

22	A total of x restrained nodal points have numbers less than or equal to zero or greater than the specified maximum value, NPOIN.
23	A total of x restrained nodal points at which the fixity code is less than or equal to zero have been detected.
24	A total of x repetitions in the list of restrained nodal points have been detected.

6.5 Standard subroutines for elasto-plastic finite element analysis

In this section we describe four additional subroutines which are common to all the elasto-plastic and elasto-viscoplastic applications presented in Chapters 7, 8 and 9. For each subroutine presented, the form of the argument list and common block structure will be that required for two-dimensional elasto-plastic applications.

6.5.1 Data input subroutine, INPUT

The role of this subroutine is to accept most of the input data required for analysis of elasto-plastic problems. The structure of this subroutine follows closely that of subroutine DATA described in Section 3.2. Subroutine INPUT also closely resembles the data input subroutine presented in Chapter 3, Ref. 4 for linear elastic problems.

The control parameters necessary for two-dimensional applications extend beyond those required for one-dimensional analysis and are presented below.

NPOIN Total number of nodal points in the structure.
NELEM Total number of elements in the structure.
NVFIX Total number of boundary points, i.e. nodal points at which one or more degrees of freedom are restrained.
NTYPE Problem type parameter:
  1—Plane stress,
  2—Plain strain,
  3—Axial symmetry.
NNODE Number of nodes per element:
  4—Linear isoparametric quadrilateral element,
  8—Quadratic isoparametric Serendipity element,
  9—Quadratic isoparametric Langrangian element.
NMATS Total number of different materials in the structure.
NGAUS The order of Gaussian quadrature rule to be employed for numerical integration of the element stiffness matrices, etc., as described in Section 6.3.2. If NGAUS is prescribed as 2 a two-point Gauss rule is to be employed; if NGAUS is input as 3 a three-point rule will be used.

NALGO Parameter controlling nonlinear solution algorithm:

1—*Initial stiffness method.* The element stiffnesses are computed at the beginning of the analysis and remain unchanged thereafter.

2—*Tangential stiffness method.* The element stiffnesses are recomputed during each iteration of each load increment.

3—*Combined algorithm.* The element stiffnesses are recomputed for the *first* iteration of each load increment only.

4—*Combined algorithm.* The element stiffnesses are recomputed for the *second* iteration of each load increment only. (Of course for the first load increment, the element stiffnesses must be calculated for the first iteration also.)

NCRIT The yield criterion to be employed:

1—Tresca,
2—Von Mises,
3—Mohr–Coulomb,
4—Drucker–Prager.

NINCS The total number of increments in which the final loading is to be applied.

NSTRE The number of independent stress components for the application:

3—Plane stress/strain,
4—Axial symmetry.

For the present two-dimensional applications two coordinate components are required to locate each nodal point. With reference to Figs. 6.2–6.4 the x, y components must be specified for plane stress or plane strain problems and the r, z components for axisymmetric situations. This information is stored in the array

COORD (IPOIN, IDIME)

where IPOIN corresponds to the number of the nodal point and IDIME refers to the coordinate component. As mentioned in Section 6.4.1 nodal coordinates need not be supplied for mid-side nodes of 8- and 9-noded elements if they lie on a straight line between corner nodes. The coordinates of such intermediate nodes are evaluated by subroutine NODEXY by linear interpolation.

For each nodal point at which the displacement value corresponding to one or more degrees of freedom are prescribed, input data must be supplied specifying these fixity conditions. The nodes at which one or more degrees of freedom are restrained are stored in array

NOFIX (IVFIX)

which signifies that the IVFIX$^{\text{th}}$ boundary node has a nodal point number NOFIX (IVFIX). Input parameter IFPRE controls which degrees of freedom of a particular node are to have a specified displacement value. For example, for plane or axisymmetric problems, integer code IFPRE may have the following values:

10 Displacement in the $x(r)$ direction specified,
01 Displacement in the $y(z)$ direction specified,
11 Displacements in both $x(r)$ and $y(z)$ directions specified.

This information is then transferred, for permanent storage, into array

IFFIX (ITOTV)

where ITOTV ranges over the total number of degrees of freedom of the structure. The prescribed displacement value associated with a restrained degree of freedom is stored in array

PRESC (IVFIX, IDOFN)

where IVFIX indicates that the prescribed displacements pertain to the IVFIX$^{\text{th}}$ boundary node and IDOFN ranges over the degrees of freedom of that node.

The list of material properties for two-dimensional applications differs from the corresponding one-dimensional case considered in Section 3.2. In particular for plane and axisymmetric elasto-plastic problems the following material parameters must be input.

PROPS (NUMAT, 1) Elastic modulus, E.
PROPS (NUMAT, 2) Poisson's ratio, ν.
PROPS (NUMAT, 3) Material thickness, t (applicable to plane problems only).
PROPS (NUMAT, 4) Material mass density, ρ.
PROPS (NUMAT, 5) Uniaxial yield stress, σ_Y (Tresca and Von Mises solids); Cohesion c (Mohr–Coulomb and Drucker–Prager materials).
PROPS (NUMAT, 6) Hardening parameter H' for linear strain hardening.
PROPS (NUMAT, 7) Angle of internal friction for Mohr–Coulomb and Drucker–Prager materials only.

Consequently NPROP $= 7$ for two-dimensional elasto-plastic applications. The corresponding material data for plate bending problems is listed in Chapter 9.

Subroutine INPUT also calls subroutine GAUSSQ, described in Section 6.4.2, whose function is to generate the sampling point position and weighting factors for numerical integration of the element stiffness matrices, etc., by Gaussian quadrature. The order of integration rule to be employed has been specified, through NGAUS, in the control data.

Subroutine INPUT is now presented and is self-explanatory.

```
      SUBROUTINE INPUT(COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,MMATS,
     .                 MPOIN,MTOTV,MVFIX,NALGO,
     .                  NCRIT,NDFRO,NDOFN,NELEM,
     .                  NEVAB,NGAUS,NGAU2,
     .                  NINCS,NMATS,NNODE,NOFIX,NPOIN,NPROP,NSTRE,NSTR1,
     .                  NTOTG,NTOTV,NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP)
C***********************************************************************
C
C**** THIS SUBROUTINE ACCEPTS MOST OF THE INPUT DATA
C
C***********************************************************************
      DIMENSION COORD(MPOIN,2),IFFIX(MTOTV),LNODS(MELEM,9),
     .          MATNO(MELEM),NDFRO(MELEM),IFPRE.7,[illegible]
     .          NOFIX(MVFIX),POSGP(4),PRESC(MVFIX,NDOFN),
     .          PROPS(MMATS,NPROP),TITLE(12),WEIGP(4)
      REWIND 1
      REWIND 2
      REWIND 3
      REWIND 4
      REWIND 8
      READ(5,920)  TITLE
      WRITE(6,920) TITLE
  920 FORMAT(12A6)
C
C*** READ THE FIRST DATA CARD, AND ECHO IT IMMEDIATELY.
C
      READ(5,900) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS,
     .NALGO,NCRIT,NINCS,NSTRE
  900 FORMAT(11I5)
      NEVAB=NDOFN*NNODE
      NSTR1=NSTRE+1
      IF(NTYPE.EQ.3) NSTR1=NSTRE
      NTOTV=NPOIN*NDOFN
      NGAU2=NGAUS*NGAUS
      NTOTG=NELEM*NGAU2
      WRITE(6,901)NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS,NEVAB,
     .NALGO,NCRIT,NINCS,NSTRE
  901 FORMAT(//8H NPOIN =,I4,4X,8H NELEM =,I4,4X,8H NVFIX =,I4,4X,
     .8H NTYPE =,I4,4X,8H NNODE =,I4,//
     . 8H NMATS =,I4,4X,8H NGAUS =,I4,
     .                 4X,8H NEVAB =,I4,4X,8H NALGO =,I4//
     . 8H NCRIT =,I4,4X,8H NINCS =,I4,4X,8H NSTRE =,I4)
      CALL       CHECK1(NDOFN,NELEM,NGAUS,NMATS,NNODE,NPOIN,
     .                  NSTRE,NTYPE,NVFIX,NCRIT,NALGO,NINCS)
C
C*** READ THE ELEMENT NODAL CONNECTIONS, AND THE PROPERTY NUMBERS.
C
      WRITE(6,902)
  902 FORMAT(//8H ELEMENT,3X,8HPROPERTY,6X,12HNODE NUMBERS)
      DO 2 IELEM=1,NELEM
      READ(5,900) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
    2 WRITE(6,903) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
  903 FORMAT(1X,I5,I9,6X,8I5)
C
C*** ZERO ALL THE NODAL COORDINATES, PRIOR TO READING SOME OF THEM.
C
      DO 4 IPOIN=1,NPOIN
      DO 4 IDIME=1,2
    4 COORD(IPOIN,IDIME)=0.0
C
C*** READ SOME NODAL COORDINATES, FINISHING WITH THE LAST NODE OF ALL.
C
      WRITE(6,904)
  904 FORMAT(//5H NODE,10X,1HX,10X,1HY)
```

```
    6 READ(5,905) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,2)
  905 FORMAT(I5,6F10.5)
      IF(IPOIN.NE.NPOIN) GO TO 6
C
C*** INTERPOLATE COORDINATES OF MID-SIDE NODES
C
      CALL      NODEXY(COORD,LNODS,MELEM,MPOIN,NELEM,NNODE)
      DO 10 IPOIN=1,NPOIN
   10 WRITE(6,906) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,2)
  906 FORMAT(1X,I5,3F10.3)
C
C*** READ THE FIXED VALUES.
C
      WRITE(6,907)
  907 FORMAT(//5H NODE,6X,4HCODE,6X,12HFIXED VALUES)
      DO 8 IVFIX=1,NVFIX
      READ(5,908) NOFIX(IVFIX),IFPRE,(PRESC(IVFIX,IDOFN),IDOFN=1,NDOFN)
      WRITE(6,908) NOFIX(IVFIX),IFPRE,(PRESC(IVFIX,IDOFN),IDOFN=1,NDOFN)
      NLOCA=(NOFIX(IVFIX)-1)*NDOFN
      IFDOF=10**(NDOFN-1)
      DO 8 IDOFN=1,NDOFN
      NGASH=NLOCA+IDOFN
      IF(IFPRE.LT.IFDOF) GO TO 8
      IFFIX(NGASH)=1
      IFPRE=IFPRE-IFDOF
    8 IFDOF=IFDOF/10
  908 FORMAT(1X,I4,5X,I5,5X,5F10.6)
C
C*** READ THE AVAILABLE SELECTION OF ELEMENT PROPERTIES.
C
   16 WRITE(6,910)
  910 FORMAT(//7H NUMBER,6X,18HELEMENT PROPERTIES)
      DO 18 IMATS=1,NMATS
      READ(5,900) NUMAT
      READ(5,930) (PROPS(NUMAT,IPROP),IPROP=1,NPROP)
  930 FORMAT(8F10.5)
   18 WRITE(6,911) NUMAT,(PROPS(NUMAT,IPROP),IPROP=1,NPROP)
  911 FORMAT(1X,I4,3X,8E14.6)
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL      GAUSSQ(NGAUS,POSGP,WEIGP)
      CALL      CHECK2(COORD,IFFIX,LNODS,MATNO,MELEM,MFRON,MPOIN,MTOTV,
     .                 MVFIX,NDFRO,NDOFN,NELEM,NMATS,NNODE,NOFIX,NPOIN,
     .                 NVFIX)
      RETURN
      END
```

6.5.2 Subroutine ALGOR

The function of this subroutine is to control the solution process according to the value of the solution algorithm parameter, NALGO, input in subroutine INPUT. This subroutine is similar in form to subroutine NONAL presented in Section 3.3 for one-dimensional applications. The subroutine sets the value of indicator KRESL to either 1 or 2 according to NALGO and the current values of the iteration number IITER and increment number IINCS. A value of KRESL = 1 indicates reformulation of the element stiffnesses accompanied by a full equation solution and KRESL = 2 indicates that the element stiffnesses are not to be modified and consequently only equation resolution takes place.

With the definitions of the permissible values of NALGO given in Section 6.5.1, subroutine ALGOR is self-explanatory and is listed below.*

```
      SUBROUTINE ALGOR(FIXED,IINCS,IITER,KRESL,
     .                 MTOTV,NALGO,NTOTV)
C***********************************************************************
C
C**** THIS SUBROUTINE SETS EQUATION RESOLUTION INDEX,KRESL
C
C***********************************************************************
      DIMENSION FIXED(MTOTV)
      KRESL=2
      IF(NALGO.EQ.1.AND.IINCS.EQ.1.AND.IITER.EQ.1) KRESL=1
      IF(NALGO.EQ.2) KRESL=1
      IF(NALGO.EQ.3.AND.IITER.EQ.1) KRESL=1
      IF(NALGO.EQ.4.AND.IINCS.EQ.1.AND.IITER.EQ.1) KRESL=1
      IF(NALGO.EQ.4.AND.IITER.EQ.2) KRESL=1
      IF(IITER.EQ.1) RETURN
      DO 100 ITOTV = 1,NTOTV
      FIXED(ITOTV)=0.0
  100 CONTINUE
      RETURN
      END
```

6.5.3 Subroutine INCREM

The role of subroutine INCREM is to increment the applied loading or any prescribed displacements according to the load factors specified as input. This subroutine is accessed on the first iteration of each load increment. For each increment of load the following items of information are input as data and are similar to those described in Section 3.7.

FACTO — This controls the magnitude of the load increment. The applied loading for each element is evaluated in Subroutine LOADPS for plane and axisymmetric situations, or Subroutine LOADPB for plate problems, and is stored in the array RLOAD (IELEM, IEVAB) as described in Section 6.4.5. The additional element load applied during the increment is RLOAD (IELEM, IEVAB)*FACTO. The applied loading is accumulative so that if FACTO is input as 0·8, 0·2 and 0·1 for the first three increments, the total load acting on the structure during the third load increment is 1·1 times the loads calculated in Subroutine LOADPS. This method of load factoring permits unequal load increments to be taken. If loading is by prescribed displacements the same factoring process holds.

TOLER — This controls the tolerance permitted on the convergence process and its use has been described in Section 3.9.3.

MITER — Maximum permissible number of iterations. This is a safety measure to cover situations where the solution process does

* For elasto-viscoplastic applications described in Chapter 8, iteration number IITER is replaced by timestep number, ISTEP.

not converge. After performing MITER iteration cycles the program will then stop.

NOUTP (1) This parameter controls the output of the unconverged results after the first iteration. In order to examine the convergence process the user can vary the frequency of output for each load increment:

1—Print the displacements only after the first iteration.
2—Print the displacements and nodal reactions after the first iteration.
3—Print the displacements, reactions and stresses after the first iteration.

NOUTP (2) This parameter controls the output of the converged results:

1—Print the final displacements only.
2—Print the final displacements and nodal reactions.
3—Print the final displacements, reactions and stresses.

The loading to which the structure is subjected is monitored by the arrays ELOAD (IELEM, IEVAB) and TLOAD (IELEM, IEVAB). The total loading applied to the structure at any stage of the analysis is accumulated in the TLOAD array. On the other hand ELOAD contains the loading to be applied to the structure for each iteration of the solution process. Initially (the first iteration of the first load increment) ELOAD contains the first increment of applied load. For the second and subsequent iterations ELOAD contains the residual nodal forces which must be redistributed as described in Section 3.7. After convergence has occurred, the next increment of load is assimilated into ELOAD, so that at this stage ELOAD contains the new applied load increment together with any residual forces still remaining after convergence of the solution for the previous load increment. These residual forces should be negligibly small if the convergence tolerance factor, TOLER, is correctly chosen. However, since any residual forces are retained in ELOAD and applied as nodal forces during the next load increment, it is noted that equilibrium is maintained at every stage of the computation process.

The final role of this subroutine is to insert appropriate values in the fixity array to control any prescribed displacements. As described in Section 3.3, in order to arrive at the correct value of a displacement whose value is prescribed for a load increment, it is necessary to prescribe the given value for equation solution during the first iteration and then prescribe a zero value for all subsequent iterations. Since the displacements occurring during each iteration accumulate to give the total displacement then clearly the prescribed value will be obtained by this process.

Subroutine INCREM will now be presented and explanatory notes provided.

```
      SUBROUTINE INCREM(ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,            INCR   1
     .                  MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NOUTP,            INCR   2
     .                  NOFIX,NTOTV,NVFIX,PRESC,RLOAD,TFACT,            INCR   3
     .                  TLOAD,TOLER)                                    INCR   4
C********************************************************************** INCR   5
C                                                                       INCR   6
C**** THIS SUBROUTINE INCREMENTS THE APPLIED LOADING                    INCR   7
C                                                                       INCR   8
C********************************************************************** INCR   9
      DIMENSION ELOAD(MELEM,MEVAB),FIXED(MTOTV),                        INCR  10
     .                  IFFIX(MTOTV),                                   INCR  11
     .          NOUTP(2),NOFIX(MVFIX),                                  INCR  12
     .          PRESC(MVFIX,NDOFN),RLOAD(MELEM,MEVAB),TLOAD(MELEM,MEVAB)INCR  13
      WRITE(6,900) IINCS                                                INCR  14
  900 FORMAT(1H0,5X,17HINCREMENT NUMBER ,I5)                            INCR  15
      READ(5,950) FACTO,TOLER,MITER,NOUTP(1),NOUTP(2)                   INCR  16
  950 FORMAT(2F10.5,3I5)                                                INCR  17
      TFACT=TFACT+FACTO                                                 INCR  18
      WRITE(6,960)TFACT,TOLER,MITER,NOUTP(1),NOUTP(2)                   INCR  19
  960 FORMAT(1H0,5X,13HLOAD FACTOR =,F10.5,5X,                          INCR  20
     .24H CONVERGENCE TOLERANCE =,F10.5,5X,24HMAX. NO. OF ITERATIONS =, INCR  21
     . I5,//27H INITIAL OUTPUT PARAMETER =,I5,5X,24HFINAL OUTPUT PARAMETINCR  22
     .ER =,I5)                                                          INCR  23
      DO 80 IELEM=1,NELEM                                               INCR  24
      DO 80 IEVAB=1,NEVAB                                               INCR  25
      ELOAD(IELEM,IEVAB)=ELOAD(IELEM,IEVAB)+RLOAD(IELEM,IEVAB)*FACTO    INCR  26
   80 TLOAD(IELEM,IEVAB)=TLOAD(IELEM,IEVAB)+RLOAD(IELEM,IEVAB)*FACTO    INCR  27
C                                                                       INCR  28
C*** INTERPRET FIXITY DATA IN VECTOR FORM                               INCR  29
C                                                                       INCR  30
      DO 100 ITOTV=1,NTOTV                                              INCR  31
  100 FIXED(ITOTV)=0.0                                                  INCR  32
      DO 110 IVFIX=1,NVFIX                                              INCR  33
      NLOCA=(NOFIX(IVFIX)-1)*NDOFN                                      INCR  34
      DO 110 IDOFN=1,NDOFN                                              INCR  35
      NGASH=NLOCA+IDOFN                                                 INCR  36
      FIXED(NGASH)=PRESC(IVFIX,IDOFN)*FACTO                             INCR  37
  110 CONTINUE                                                          INCR  38
      RETURN                                                            INCR  39
      END                                                               INCR  40
```

INCR 14–15 Write the number of the load increment which is being currently solved.

INCR 16–23 Read and write the load increment control parameters. Note that the incremental load factor, FACTO, is input whereas the *total* load factor, TFACT, is output.

INCR 24–27 Accumulate the incremental loading into array ELOAD for equation solution and also into TLOAD to record the total load applied to the structure.

INCR 31–32 Zero the global vector of prescribed displacements.

INCR 33–38 Insert any prescribed displacement values, factored by the load increment factor, into the appropriate position in the global vector.

6.5.4 Solution convergence monitoring subroutine CONVER

This subroutine monitors convergence of the nonlinear solution iteration process. It is almost identical to subroutine CONUND for one-dimensional applications described in Section 3.10.3. Since for two-dimensional and plate bending problems we have more than one degree of freedom per nodal point, summation in (3.27) must now be made over the total number of degrees of freedom in the structure. As an additional check on the nonlinear solution process we also arrange to evaluate the maximum individual residual force ψ_i^r existing in the structure.

Subroutine CONVER is now presented and can be understood with the aid of Section 3.10.3.

```
      SUBROUTINE CONVER(ELOAD,IITER,LNODS,MELEM,MEVAB,MTOTV,NCHEK,
     .                  NDOFN,NELEM,NEVAB,NNODE,NTOTV,PVALU,STFOR,
     .                  TLOAD,TOFOR,TOLER)
C*********************************************************************
C
C**** THIS SUBROUTINE CHECKS FOR CONVERGENCE OF THE ITERATION PROCESS
C
C*********************************************************************
      DIMENSION ELOAD(MELEM,MEVAB),LNODS(MELEM,9) ,STFOR(MTOTV),
     .          TOFOR(MTOTV),TLOAD(MELEM,MEVAB)
      NCHEK=0
      RESID=0.0
      RETOT=0.0
      REMAX=0.0
      DO 5 ITOTV=1,NTOTV
      STFOR(ITOTV)=0.0
      TOFOR(ITOTV)=0.0
    5 CONTINUE
      DO 40 IELEM=1,NELEM
      KEVAB=0
      DO 40 INODE=1,NNODE
      LOCNO=IABS(LNODS(IELEM,INODE))
      DO 40 IDOFN=1,NDOFN
      KEVAB=KEVAB+1
      NPOSI=(LOCNO-1)*NDOFN+IDOFN
      STFOR(NPOSI)=STFOR(NPOSI)+ELOAD(IELEM,KEVAB)
   40 TOFOR(NPOSI)=TOFOR(NPOSI)+TLOAD(IELEM,KEVAB)
      DO 50 ITOTV=1,NTOTV
      REFOR=TOFOR(ITOTV)-STFOR(ITOTV)
      RESID=RESID+REFOR*REFOR
      RETOT=RETOT+TOFOR(ITOTV)*TOFOR(ITOTV)
      AGASH=ABS(REFOR)
   50 IF(AGASH.GT.REMAX) REMAX=AGASH
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=TLOAD(IELEM,IEVAB)-ELOAD(IELEM,IEVAB)
      RESID=SQRT(RESID)
      RETOT=SQRT(RETOT)
      RATIO=100.0*RESID/RETOT
      IF(RATIO.GT.TOLER) NCHEK=1
      IF(IITER.EQ.1) GO TO 20
      IF(RATIO.GT.PVALU) NCHEK=999
   20 PVALU=RATIO
      WRITE(6,30) NCHEK,RATIO,REMAX
   30 FORMAT(1H0,3X,18HCONVERGENCE CODE =,I4,3X,28HNORM OF RESIDUAL SUM
     .RATIO =,E14.6,3X,18HMAXIMUM RESIDUAL =,E14.6)
      RETURN
      END
```

6.6 Problems

6.1 Using the subroutines described in this chapter devise programs to evaluate the stiffness matrices and load vectors for 4-, 8- and 9-node quadrilateral isoparametric elements for plane stress, plane strain, axisymmetric and Mindlin plate applications.

6.2 Use the shape functions $L_i^{(e)}(\xi, \eta)$ from the 9-node Lagrangian quadrilateral isoparametric element to devise a new family of 8-node Serendipity quadrilateral element shape functions $N_i^{(e)}(\xi, \eta)$ of the form

$$N_i^{(e)} = L_i^{(e)} + aL_9^{(e)} \qquad i = 1, 3, 5 \text{ and } 7 \text{ (corner nodes)},$$

$$N_i^{(e)} = L_i^{(e)} + bL_9^{(e)} \qquad i = 2, 4, 6 \text{ and } 8 \text{ (midside nodes)},$$

where $L_9^{(e)}$ is the shape function of the central node of the Lagrangian element. What limits are there on a and b?

6.3 Determine some further diagnostic checks on the input, other than those described in Sections 6.4.13 and 6.4.15. Apart from the check on the Jacobian determinant given in Subroutine JACOB2 in Section 6.4.4, are there any other checks which could be incorporated into the program after the input has been successfully read and checked?

6.4 Determine the consistent nodal forces for the case when a point load with components P_x, P_y acts at an arbitrary point along an element edge defined by Cartesian coordinates (x_P, y_P), which correspond to local coordinates $(\xi, \eta) = (\xi_P, -1)$.

6.7 References

1. Hill, R., *The Mathematical Theory of Plasticity*, Oxford University Press, 1950.
2. Timoshenko, S. P. and Goodier, J. N., *Theory of Elasticity*, McGraw-Hill, New York, 1951.
3. Hughes, T. J. R., Cohen, M. and Haroun, M., Reduced and selective integration techniques in the finite element analysis of plates, *Nucl. Eng. Design*, **46**, 203–222 (1978).
4. Hinton, E. and Owen, D. R. J., *Finite Element Programming*, Academic Press, London, 1977.
5. Irons, B. M. and Ahmad, S., *Techniques of Finite Elements*, Ellis Horwood, Chichester, 1980.
6. Bathe, K. J. and Wilson, E. L., *Numerical Methods in Finite Element Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, 1977.

Chapter 7
Elasto-plastic problems in two dimensions

7.1 Introduction

In this chapter we consider the elasto-plastic stress analysis of solids which conform to plane stress, plane strain or axisymmetric conditions. Most of the problems encountered in engineering can be approximated to satisfy one of these classifications.

The basic laws governing elasto-plastic material behaviour in a two-dimensional solid must be presented before the numerical aspects of the problem can be considered and to this end new concepts, such as the plastic potential and the normality condition will be introduced. Only the essential expressions will be provided in this text and the reader will be directed to other sources for a more complete theoretical treatment.

The situation is complicated by the fact that different classes of materials exhibit different elasto-plastic characteristics. In this chapter four different yield criteria are employed. The Tresca and Von Mises laws, which closely approximate metal plasticity behaviour, are considered and the Mohr–Coulomb and Drucker–Prager criteria, which are applicable to concrete, rocks and soils, are presented.

In the latter sections of this chapter a computer code is developed to allow the solution of practical problems. Many of the subroutines required for elasto-plastic solution have been reviewed in Chapter 6. In this chapter the additional subroutines are developed and assembled to provide a working program.

7.2 The mathematical theory of plasticity

The object of the mathematical theory of plasticity is to provide a theoretical description of the relationship between stress and strain for a material which exhibits an elasto-plastic response. In essence, plastic behaviour is characterised by an irreversible straining which is not time dependent and which can only be sustained once a certain level of stress has been reached. In this section we outline the basic assumptions and associated theoretical expressions for a general continuum. For a more complete treatment the reader is directed to Refs. 1–3. In order to formulate a theory which models elasto-plastic material deformation three requirements have to be met:

- An explicit relationship between stress and strain must be formulated to describe material behaviour under elastic conditions, i.e. before the onset of plastic deformation.
- A yield criterion indicating the stress level at which plastic flow commences must be postulated.
- A relationship between stress and strain must be developed for post-yield behaviour, i.e. when the deformation is made up of both elastic and plastic components.

Before the onset of plastic yielding the relationship between stress and strain is given by the standard linear elastic expression.*

$$\sigma_{ij} = C_{ijkl}\,\epsilon_{kl}, \tag{7.1}$$

where σ_{ij} and ϵ_{kl} are the stress and strain components respectively and C_{ijkl} is the tensor of elastic constants which for an isotropic material has the explicit form

$$C_{ijkl} = \lambda\,\delta_{ij}\delta_{kl}+\mu\,\delta_{ik}\delta_{jl}+\mu\,\delta_{il}\delta_{jk}, \tag{7.2}$$

where λ and μ are the Lamé constants and δ_{ij} is the Kronecker delta defined by

$$\delta_{ij} = \begin{cases} 1 & \text{if} \quad i=j \\ 0 & \quad\;\; i \neq j. \end{cases} \tag{7.3}$$

7.2.1 The yield criterion

The yield criterion determines the stress level at which plastic deformation begins and can be written in the general form

$$f(\sigma_{ij}) = k(\kappa), \tag{7.4}$$

where f is some function and k a material parameter to be determined experimentally. The term k may be a function of a hardening parameter κ discussed later in Section 7.2.2. On physical grounds, any yield criterion should be independent of the orientation of the coordinate system employed and therefore it should be a function of the three stress invariants only

$$\begin{aligned} J_1 &= \sigma_{ii} \\ J_2 &= \tfrac{1}{2}\sigma_{ij}\sigma_{ij} \\ J_3 &= \tfrac{1}{3}\sigma_{ij}\sigma_{jk}\sigma_{ki}. \end{aligned} \tag{7.5}$$

Experimental observations, notably by Bridgeman,[4] indicate that plastic deformation of metals is essentially independent of hydrostatic pressure. Consequently the yield function can only be of the form

$$f(J_2', J_3') = k(\kappa), \tag{7.6}$$

* In the indicial notation employed, Einstein's summation convention is invoked, whereby it is implicitly assumed that a summation from 1 to 3 is performed over any index which is repeated in any term of an expression. Also indices 1, 2, 3 refer to Cartesian components x, y, z respectively. Note that $\sigma_{11}=\sigma_{xx}=\sigma_x$, $\sigma_{12}=\sigma_{xy}$, etc.

where J_2' and J_3' are the second and third invariants of the deviatoric stresses,

$$\sigma_{ij}' = \sigma_{ij} - \tfrac{1}{3}\delta_{ij}\,\sigma_{kk}. \tag{7.7}$$

Most of the various yield criteria that have been suggested for metals are now only of historic interest, since they conflict with experimental predictions. The two simplest which do not have this fault are the Tresca criterion and the Von Mises criterion.

The Tresca yield criterion (1864)

This states that yielding begins when the maximum shear stress reaches a certain value. If the principal stresses are σ_1, σ_2, σ_3 where $\sigma_1 \geqslant \sigma_2 \geqslant \sigma_3$ then yielding begins when

$$\sigma_1 - \sigma_3 = Y(\kappa), \tag{7.8}$$

where Y is a material parameter to be experimentally determined and which may be a function of the hardening parameter κ. By considering all other possible maximum shearing stress values (e.g. $\sigma_2 - \sigma_1$ if $\sigma_2 \geqslant \sigma_3 \geqslant \sigma_1$) it can be shown that this yield criterion may be represented in the $\sigma_1\sigma_2\sigma_3$ *stress space* by the surface of an infinitely long regular hexagonal cylinder as shown in Fig. 7.1. The axis of the cylinder coincides with the *space diagonal*, defined by points $\sigma_1 = \sigma_2 = \sigma_3$, and since each normal section of the cylinder is identical, (a consequence of the assumption that a hydrostatic stress does not influence yielding), it is convenient to represent the *yield surface* geometrically by projecting it onto the so-called π plane, $\sigma_1+\sigma_2+\sigma_3 = 0$ as shown in Fig. 7.2(a). When the yield function f depends on J_2' and J_3' alone it can be

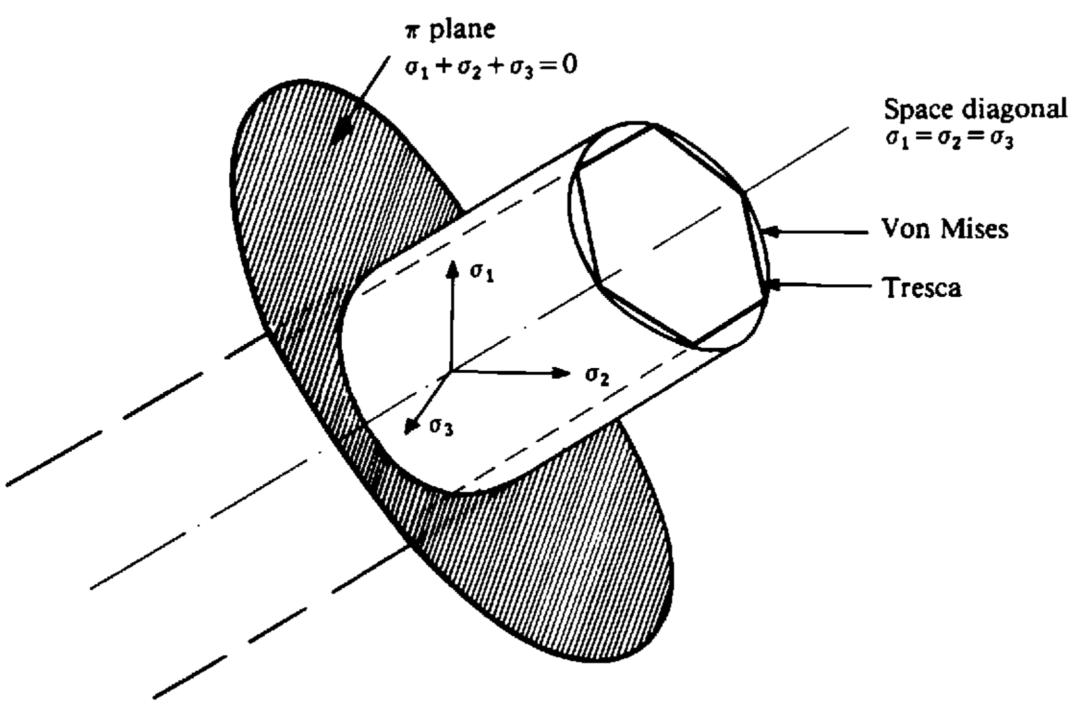


Fig. 7.1 Geometrical representation of the Tresca and Von Mises yield surfaces in principal stress space.

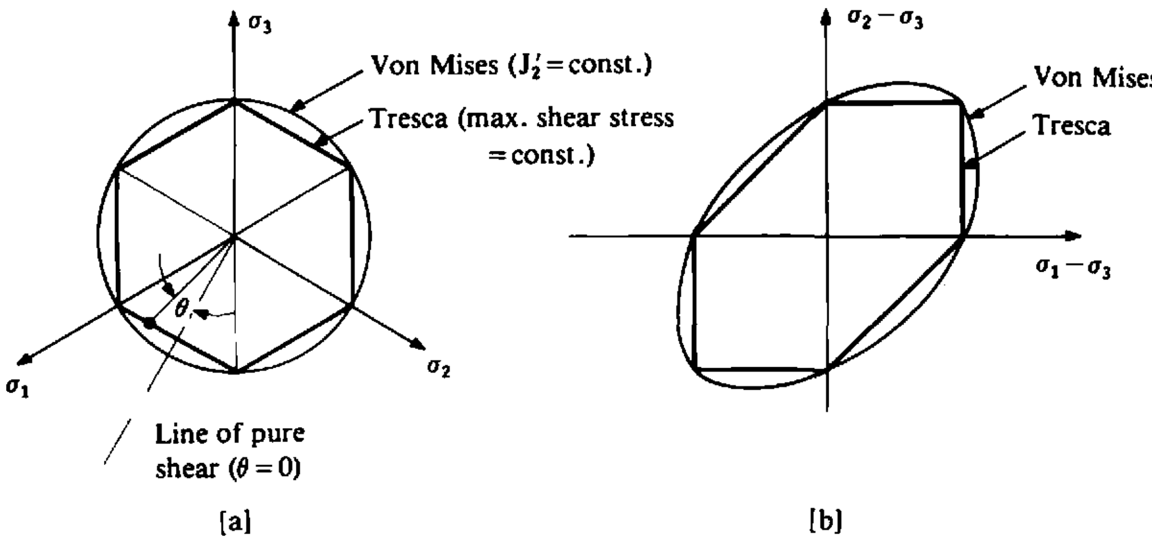


Fig. 7.2 Two-dimensional representations of the Tresca and Von Mises yield criteria. (a) π plane representation. (b) Conventional engineering representation.

written in the form $f(\sigma_1-\sigma_3, \sigma_2-\sigma_3)$ and a two-dimensional plot of the surface $f = k$ is then possible as shown in Fig. 7.2(b). It can be shown generally [1,2] that yield surfaces must be convex (except for local flat areas, possibly) and that they must contain the stress origin.

The Von Mises yield criterion (1913)

Von Mises suggested that yielding occurs when J_2' reaches a critical value, or

$$(J_2')^{\frac{1}{2}} = k(\kappa), \tag{7.9}$$

in which k is a material parameter to be determined. The second deviatoric stress invariant, J_2', can be explicitly written as

$$\begin{aligned} J_2' &= \tfrac{1}{2}\sigma_{ij}'\sigma_{ij}' = \tfrac{1}{6}[(\sigma_1-\sigma_2)^2+(\sigma_2-\sigma_3)^2+(\sigma_3-\sigma_1)^2] \\ &= \tfrac{1}{2}[\sigma_x'^2+\sigma_y'^2+\sigma_z'^2]+\tau_{xy}^2+\tau_{yz}^2+\tau_{xz}^2. \end{aligned} \tag{7.10}$$

Yield criterion (7.9) may be further written as

$$\bar{\sigma} = \sqrt{3}(J_2')^{\frac{1}{2}} = \sqrt{3}k, \tag{7.11}$$

where

$$\bar{\sigma} = \sqrt{(3/2)}\,\{\sigma_{ij}'\sigma_{ij}'\}^{\frac{1}{2}}, \tag{7.12}$$

and $\bar{\sigma}$ is termed the *effective stress, generalised stress* or *equivalent stress*. Some physical insight into the definition of $\bar{\sigma}$ will be apparent later from Section 7.2.4 where the case of uniaxial yielding is considered. There are two physical interpretations of the Von Mises yield condition. Nadai (1937) introduced the so-called *octahedral shear stress* τ_{oct}, which is the shear stress on the planes of a regular octahedron, the apices of which coincide with the principal axes of stress. The value of τ_{oct} is related to J_2' by

$$\tau_{oct} = \sqrt{(2J_2'/3)}. \tag{7.13}$$

Thus yielding can be interpreted to begin when τ_{oct} reaches a critical value. Hencky (1924) pointed out that the Von Mises law implies that yielding begins when the (recoverable) elastic energy of distortion reaches a critical value.

Fig. 7.1 shows the geometrical interpretation of the Von Mises yield surface to be a circular cylinder whose projection onto the π plane is a circle of radius $\sqrt{(2)}k$ as shown in Fig. 7.2(a). The two dimensional plot of the Von Mises yield surface is the ellipse shown in Fig. 7.2(b). A physical meaning of the constant k can be obtained by considering the yielding of materials under simple stress states. The case of pure shear ($\sigma_1 = -\sigma_2$, $\sigma_3 = 0$) requires on use of (7.9) and (7.10) that k must equal the yield shear stress. Alternatively the case of uniaxial tension ($\sigma_2 = \sigma_3 = 0$) requires that $\sqrt{(3)}k$ is the uniaxial yield stress.

The Tresca yield locus is a hexagon with distances of $\sqrt{(2/3)}\,Y$ from origin to apex on the π plane whereas the Von Mises yield surface is a circle of radius $\sqrt{(2)}k$. By suitably choosing the constant Y, the criteria can be made to agree with each other, and with experiment, for a single state of stress. This may be selected arbitrarily; it is conventional to make the circle pass through the apices of the hexagon by taking the constant $Y = \sqrt{(3)}k$, the yield stress in simple tension. The criteria then differ most for a state of pure shear, where the Von Mises criterion gives a yield stress $2/\sqrt{(3)}$ ($\approx 1{\cdot}15$) times that given by the Tresca criterion. For most metals Von Mises' law fits the experimental data more closely than Tresca's, but it frequently happens that the Tresca criterion is simpler to use in theoretical applications.

The Mohr–Coulomb yield criterion

This is a generalisation of the Coulomb (1773) friction failure law defined by

$$\tau = c - \sigma_n \tan\phi, \tag{7.14}$$

where τ is the magnitude of the shearing stress, σ_n is the normal stress (tensile stress is positive), c is the cohesion and ϕ the angle of internal friction. Graphically (7.14) represents a straight line tangent to the largest principal stress circle as shown in Fig. 7.3 and was first demonstrated by Mohr (1882). From Fig. 7.3, and for $\sigma_1 \geqslant \sigma_2 \geqslant \sigma_3$ (7.14) can be rewritten as

$$-\tfrac{1}{2}(\sigma_1 - \sigma_3)\cos\phi = c - \left(\frac{\sigma_1 + \sigma_3}{2} - \frac{(\sigma_1 - \sigma_3)}{2}\sin\phi\right)\tan\phi, \tag{7.15}$$

or rearranging

$$(\sigma_1 - \sigma_3) = 2c\cos\phi - (\sigma_1 + \sigma_3)\sin\phi. \tag{7.16}$$

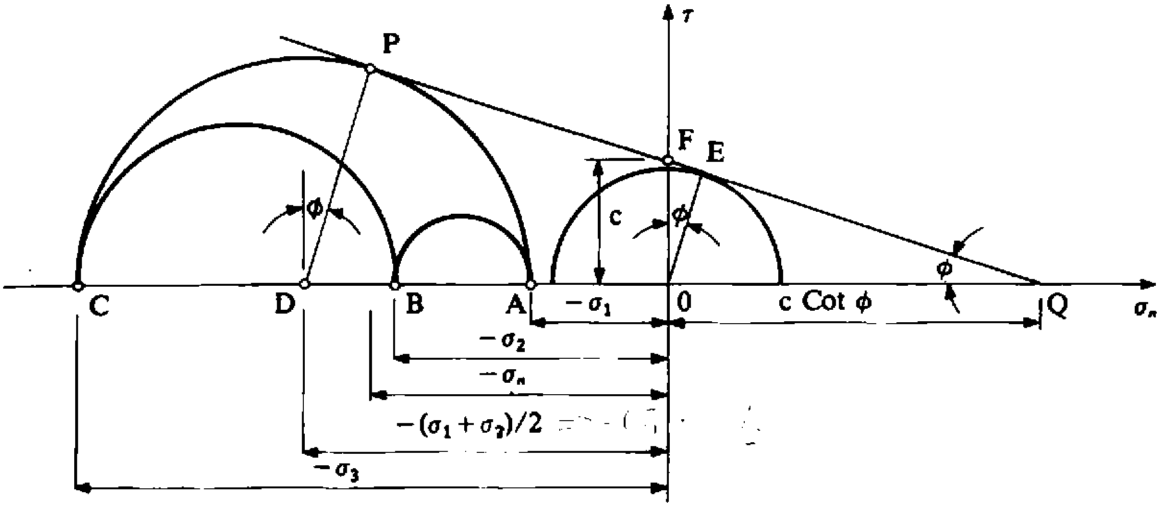


Fig. 7.3 Mohr circle representation of the Mohr–Coulomb yield criterion.

Again, as for the Tresca criterion, the complete yield surface is obtained by considering all other stress combinations which can cause yielding (e.g. $\sigma_3 \geqslant \sigma_1 \geqslant \sigma_2$). In principal stress space this gives a conical yield surface whose normal section at any point is an irregular hexagon as shown in Fig. 7.4. The conical, rather than cylindrical, nature of the yield surface is a consequence of the fact that a hydrostatic stress does influence yielding which is evident from the last term in (7.14). When $\sigma_1 = \sigma_2 = \sigma_3$ we have from (7.16) that the mean hydrostatic stress, $\sigma_m = c \cot\phi$ and therefore the apex of the hexagonal pyramid, 0, in Fig. 7.4, lies along the space diagonal at the point $\sigma_1 = \sigma_2 = \sigma_3 = c \cot\phi$. This criterion is applicable to concrete, rock and soil problems.

The Drucker–Prager yield criterion

An approximation to the Mohr–Coulomb law was presented by Drucker and Prager (1952) as a modification of the Von Mises yield criterion. The influence of a hydrostatic stress component on yielding was introduced by inclusion of an additional term in the Von Mises expression to give

$$\alpha J_1 + (J_2')^{\frac{1}{2}} = k'. \tag{7.17}$$

This yield surface has the form of a circular cone. In order to make the Drucker–Prager circle coincide with the outer apices of the Mohr–Coulomb hexagon at any section, it can be shown that

$$\alpha = \frac{2\sin\phi}{\sqrt{(3)}(3-\sin\phi)}, \quad k' = \frac{6c\cos\phi}{\sqrt{(3)}(3-\sin\phi)}. \tag{7.18}$$

Coincidence with the inner apices of the Mohr–Coulomb hexagon is provided by

$$\alpha = \frac{2\sin\phi}{\sqrt{(3)}(3+\sin\phi)}, \quad k' = \frac{6c\cos\phi}{\sqrt{(3)}(3+\sin\phi)}. \tag{7.19}$$

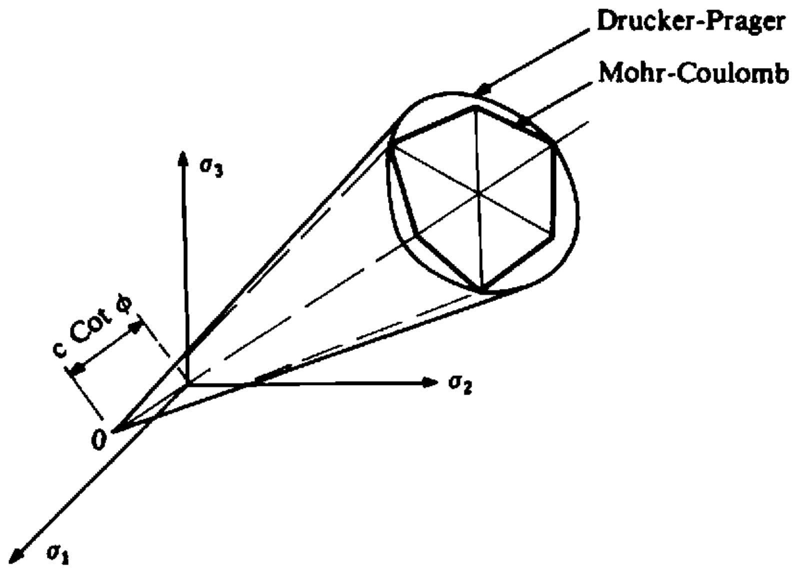


Fig. 7.4 (a) Geometrical representation of the Mohr–Coulomb and Drucker–Prager yield surfaces in principal stress space.

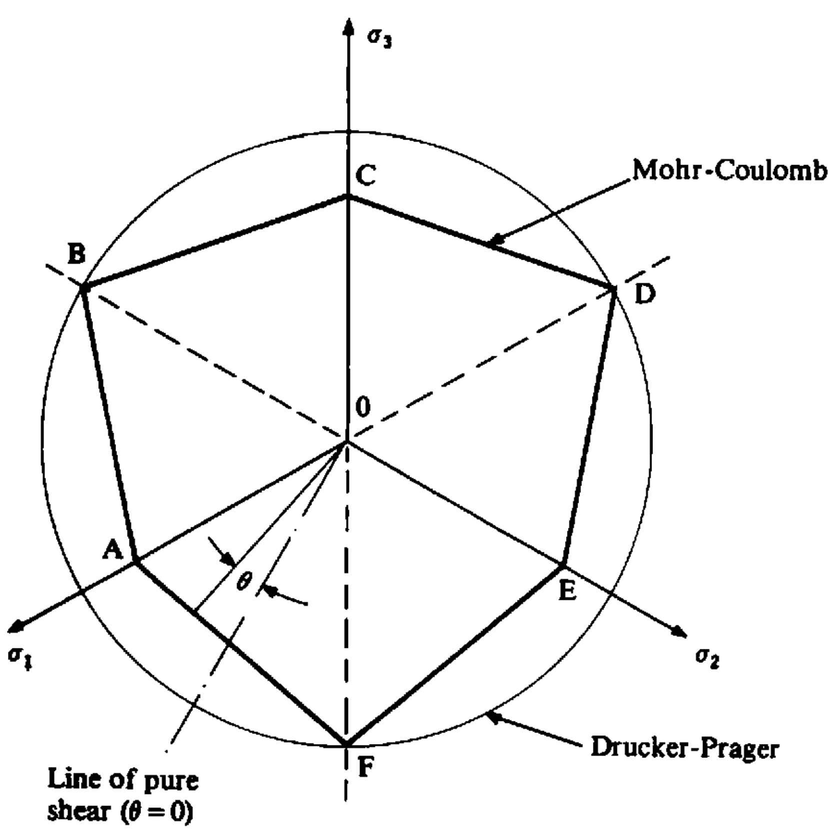
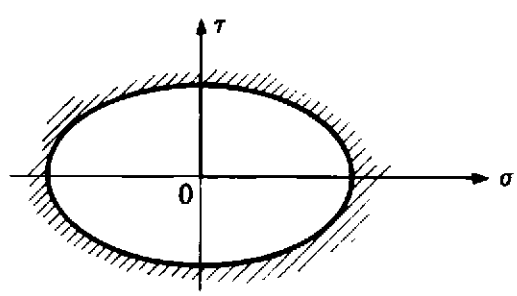


Fig. 7.4 (b) Two-dimensional, π plane, representation of the Mohr–Coulomb and Drucker–Prager yield criteria.

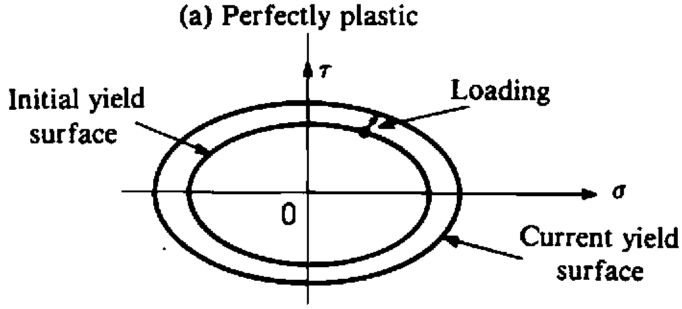
However, the approximation given by either the inner or outer cone to the true failure surface can be poor for certain stress combinations.[5]

7.2.2 Work or strain hardening

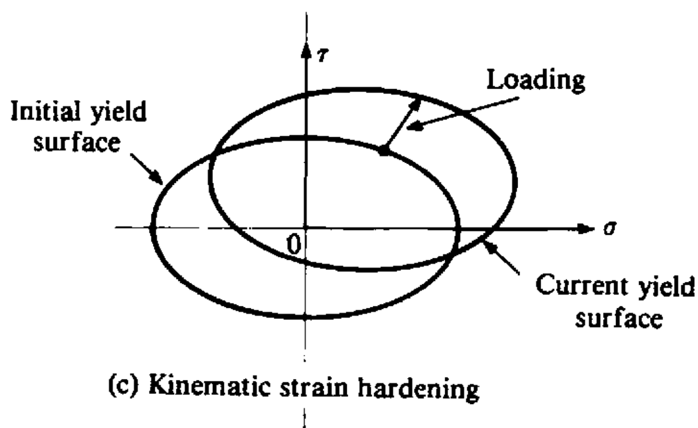
After initial yielding, the stress level at which further plastic deformation occurs may be dependent on the current degree of plastic straining. Such a phenomenon is termed work hardening or strain hardening. Thus the yield surface will vary at each stage of the plastic deformation, with the subsequent yield surfaces being dependent on the plastic strains in some way. Some alternative models which describe strain hardening in a material are illustrated in Fig. 7.5. A perfectly plastic material is shown in Fig. 7.5(a) where the yield stress level does not depend in any way on the degree of plastification. If the subsequent yield surfaces are a uniform expansion of the original yield curve, without translation, as shown in Fig. 7.5(b) the strain-hardening model is said to be *isotropic*. On the other hand if the subsequent yield surfaces preserve their shape and orientation but translate in the stress space as a rigid body as shown in Fig. 7.5(c), *kinematic* hardening is said to take place. Such a hardening model gives rise to the experimentally observed Bauschinger effect on cyclic loading.



(a) Perfectly plastic



(b) Isotropic strain hardening



(c) Kinematic strain hardening

Fig. 7.5 Mathematical models for representation of strain hardening behaviour.

For some materials, notably soils, the yield surface may not strain harden but *strain soften* instead, so that the yield stress level at a point decreases with increasing plastic deformation. Therefore, for an isotropic model, the original yield curve contracts progressively without translation. Consequently yielding implies local failure and the yield surface becomes a *failure criterion*.

The progressive development of the yield surface can be defined by relating the yield stress k to the plastic deformation by means of the hardening parameter κ. This can be done in two ways. Firstly the degree of *work hardening* can be postulated to be a function of the total plastic work, W_p, only. Then,

$$\kappa = W_p, \tag{7.20}$$

where

$$W_p = \int \sigma_{ij}(d\epsilon_{ij})_p, \tag{7.21}$$

in which $(d\epsilon_{ij})_p$ are the plastic components of strain occurring during a strain increment. Alternatively κ can be related to a measure of the total plastic deformation termed the *effective, generalised or equivalent plastic strain* which is defined incrementally as

$$d\bar{\epsilon}_p = \sqrt{(\tfrac{2}{3})}\{(d\epsilon_{ij})_p(d\epsilon_{ij})_p\}^{\frac{1}{2}}. \tag{7.22}$$

A physical insight of this definition is provided in Section 7.2.4 where uni-axial yielding is considered. For situations where the assumption that yielding is independent of any hydrostatic stress is valid, $(d\epsilon_{ii})_p = 0$ and hence $(d\epsilon_{ij}')_p = (d\epsilon_{ij})_p$. Consequently (7.22) can be rewritten as

$$d\bar{\epsilon}_p = \sqrt{(\tfrac{2}{3})}\{(d\epsilon_{ij}')_p(d\epsilon_{ij}')_p\}^{\frac{1}{2}}. \tag{7.23}$$

Then the hardening parameter, κ, is assumed to be defined as

$$\kappa = \bar{\epsilon}_p, \tag{7.24}$$

where $\bar{\epsilon}_p$ is the result of integrating $d\bar{\epsilon}_p$ over the strain path. This behaviour is termed *strain hardening*. Only an isotropic hardening model will be considered in this text.

Stress states for which $f = k$ represent plastic states, while elastic behaviour is characterised by $f<k$. At a plastic state, $f = k$, the incremental change in the yield function due to an incremental stress change is

$$df = \frac{\partial f}{\partial \sigma_{ij}} d\sigma_{ij}. \tag{7.25}$$

Then if:-

- $df<0$ elastic unloading occurs (elastic behaviour) and the stress point returns inside the yield surface
- $df=0$ neutral loading (plastic behaviour for a perfectly plastic material) and the stress point remains on the yield surface

$df > 0$ plastic loading (plastic behaviour for a strain hardening material) and the stress point remains on the expanding yield surface.

It can also be shown[1-3] that, for a stable material that the initial and all subsequent yield surfaces must be convex.

7.2.3 Elasto-plastic stress/strain relation

After initial yielding the material behaviour will be partly elastic and partly plastic. During any increment of stress, the changes of strain are assumed to be divisible into elastic and plastic components, so that

$$d\epsilon_{ij} = (d\epsilon_{ij})_e + (d\epsilon_{ij})_p. \tag{7.26}$$

The elastic strain increment is related to the stress increment by (7.1). Or, decomposing the stress terms into their deviatoric and hydrostatic components

$$(d\epsilon_{ij})_e = \frac{d\sigma_{ij}'}{2\mu} + \frac{(1-2\nu)}{E}\delta_{ij}d\sigma_{kk}, \tag{7.27}$$

where E and ν are respectively the elastic modulus and Poisson's ratio of the material.

In order to derive the relationship between the plastic strain component and the stress increment a further assumption on the material behaviour must be made. In particular it will be assumed that, the plastic strain increment is proportional to the stress gradient of a quantity termed the *plastic potential* Q, so that

$$(d\epsilon_{ij})_p = d\lambda\frac{\partial Q}{\partial\sigma_{ij}}, \tag{7.28}$$

where $d\lambda$ is a proportionality constant termed the *plastic multiplier*. A theoretical basis for this assumption is developed in Ref. 1. Equation (7.28) is termed the *flow rule* since it governs the plastic flow after yielding. The potential Q must be a function of J_2' and J_3' but as yet it cannot be determined in its most general form. However the relation $f \equiv Q$ has a special significance in the mathematical theory of plasticity, since for this case certain variational principles and uniqueness theorems can be formulated. The identity $f \equiv Q$ is a valid one since it has been postulated that both are functions of J_2' and J_3' and such an assumption gives rise to an *associated* theory of plasticity. In this case (7.28) becomes

$$(d\epsilon_{ij})_p = d\lambda\frac{\partial f}{\partial\sigma_{ij}}, \tag{7.29}$$

and is termed the *normality condition* since $\partial f/\partial\sigma_{ij}$ is a vector directed normal to the yield surface at the stress point under consideration as shown in Fig. 7.6. It is seen that the components of the plastic strain increment are required to combine vectorially in n-dimensional space to give a vector

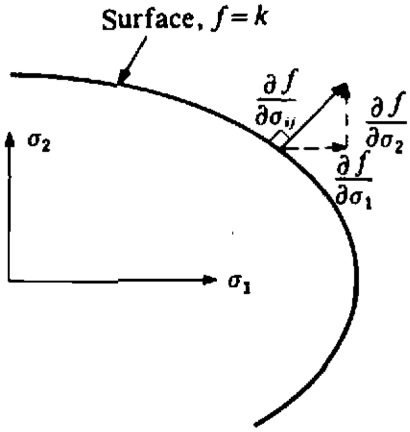


Fig. 7.6 Geometrical representation of the normality rule of associated plasticity.

which is normal to the yield surface. For the particular case of $f = J_2'$ we have

$$\frac{\partial f}{\partial \sigma_{ij}} = \frac{\partial J_2'}{\partial \sigma_{ij}} = \sigma_{ij}'. \tag{7.30}$$

Then (7.29) becomes

$$(d\epsilon_{ij})_p = d\lambda \sigma_{ij}', \tag{7.31}$$

which are known as the *Prandtl–Reuss equations*[1] and have been extensively employed in theoretical work. Experimental observations indicate that the normality condition is an acceptable assumption for metals, but the question of normality in rocks and soils is still open to debate[6] and is discussed further in Chapter 12. Thus on use of (7.26), (7.27) and (7.29) the complete incremental relationship between stress and strain for elasto-plastic deformation is found to be

$$d\epsilon_{ij} = \frac{d\sigma_{ij}'}{2\mu} + \frac{(1-2\nu)}{E}\delta_{ij}d\sigma_{kk} + d\lambda\frac{\partial f}{\partial \sigma_{ij}}. \tag{7.32}$$

7.2.4 Uniaxial yield test on a strain-hardening material

Consider the uniaxial testing of an elasto-plastic material which produces the stress-strain curve shown in Fig. 7.7. The behaviour is initially elastic characterised by an elastic modulus E until yielding commences at the uniaxial yield stress σ_Y. Thereafter the material response is elasto-plastic with the local tangent to the curve continually varying and is termed *the elasto-plastic tangent modulus*, E_T. The hardening law $k = k(\kappa)$ could just as easily be expressed in terms of the effective stress, $\bar{\sigma}$ (since it is proportional to J_2') to give, for the strain hardening hypothesis (7.24)

$$\bar{\sigma} = H(\bar{\epsilon}_p), \tag{7.33}$$

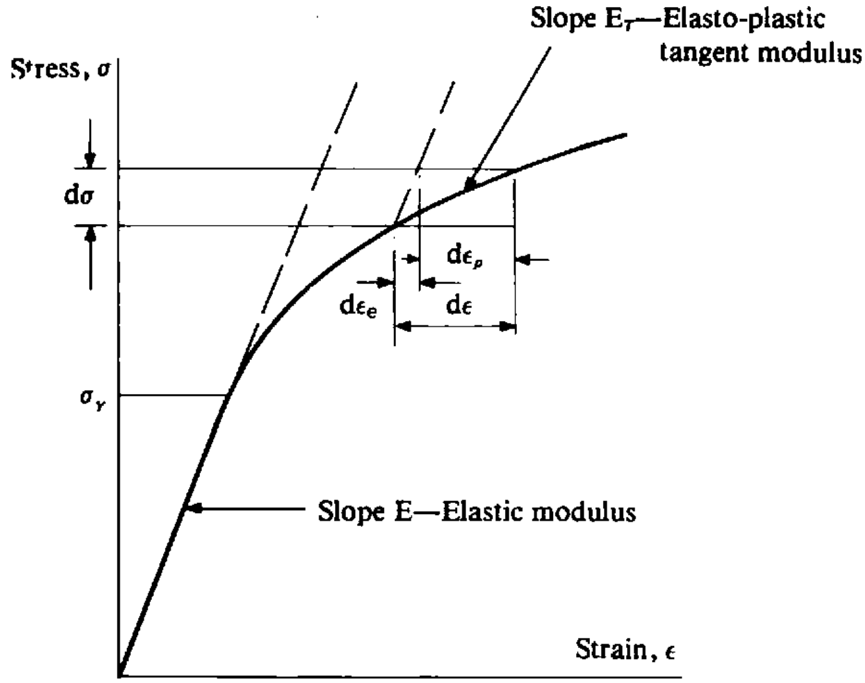


Fig. 7.7 Elasto-plastic strain hardening behaviour for the uniaxial case.

or differentiating,

$$\frac{d\bar{\sigma}}{d\bar{\epsilon}_p} = H'(\bar{\epsilon}_p). \tag{7.34}$$

For the uniaxial case under consideration $\sigma_1 = \sigma$, $\sigma_2 = \sigma_3 = 0$ and thus from (7.12)

$$\bar{\sigma} = \sqrt{(\tfrac{3}{2})}\{\sigma_{ij}'\sigma_{ij}'\}^{1/2} = \sigma. \tag{7.35}$$

If the plastic strain increment in the direction of loading is $d\epsilon_p$, then $(d\epsilon_1)_p = d\epsilon_p$ and since plastic straining is assumed to be incompressible, Poisson's ratio is effectively 0.5 and $(d\epsilon_2)_p = -\frac{1}{2}d\epsilon_p$ and $(d\epsilon_3)_p = -\frac{1}{2}d\epsilon_p$. Then from (7.23) the effective plastic strain becomes

$$d\bar{\epsilon}_p = \sqrt{(\tfrac{2}{3})}\{(\epsilon_{ij}')_p(\epsilon_{ij}')_p\}^{1/2} = d\epsilon_p. \tag{7.36}$$

Expressions (7.35) and (7.36) explain the apparent arbitrary constants employed in the definition of $\bar{\sigma}$ and $\bar{\epsilon}_p$, since these terms are required to become the actual stress and strain for uniaxial yielding. Using (7.35) and (7.36) then (7.34) becomes

$$H'(\bar{\epsilon}_p) = \frac{d\sigma}{d\epsilon_p} = \frac{d\sigma}{d\epsilon - d\epsilon_e} = \frac{1}{d\epsilon/d\sigma - d\epsilon_e/d\sigma},$$

or

$$H' = \frac{E_T}{1 - E_T/E}. \tag{7.37}$$

Thus the hardening function H' can be determined experimentally from a simple uniaxial yield test. (For numerical computation it will be shown in the next section that it is H' and not H that is required).

7.3 Matrix formulation

The theoretical expressions developed in Section 7.2 will now be converted to matrix form.[7,8] The yield function, first defined in (7.4), can be rewritten as

$$f(\boldsymbol{\sigma}) = k(\kappa), \tag{7.38}$$

where $\boldsymbol{\sigma}$ is the stress vector and κ is the hardening parameter which governs the expansion of the yield surface. In particular, from (7.20) and (7.21), $d\kappa = \boldsymbol{\sigma}^T d\boldsymbol{\epsilon}_p$ for the work hardening hypothesis and from (7.24) $d\kappa = d\bar{\epsilon}_p$ for the strain hardening hypothesis. Rearranging (7.38) we get

$$F(\boldsymbol{\sigma},\kappa) = f(\boldsymbol{\sigma}) - k(\kappa) = 0. \tag{7.39}$$

By differentiating (7.39) we have

$$dF = \frac{\partial F}{\partial \boldsymbol{\sigma}} d\boldsymbol{\sigma} + \frac{\partial F}{\partial \kappa} d\kappa = 0, \tag{7.40}$$

or

$$\boldsymbol{a}^T d\boldsymbol{\sigma} - A\, d\lambda = 0, \tag{7.41}$$

with the definitions

$$\boldsymbol{a}^T = \frac{\partial F}{\partial \boldsymbol{\sigma}} = \left[\frac{\partial F}{\partial \sigma_x}, \frac{\partial F}{\partial \sigma_y}, \frac{\partial F}{\partial \sigma_z}, \frac{\partial F}{\partial \tau_{yz}}, \frac{\partial F}{\partial \tau_{zx}}, \frac{\partial F}{\partial \tau_{xy}}\right], \tag{7.42}$$

and

$$A = -\frac{1}{d\lambda} \frac{\partial F}{\partial \kappa} d\kappa. \tag{7.43}$$

The vector $\boldsymbol{a}$ is termed the *flow vector*. Expression (7.32) can be immediately rewritten as

$$d\boldsymbol{\epsilon} = [\boldsymbol{D}]^{-1} d\boldsymbol{\sigma} + d\lambda \frac{\partial F}{\partial \boldsymbol{\sigma}}, \tag{7.44}$$

where $\boldsymbol{D}$ is the usual matrix of elastic constants. Premultiplying both sides of (7.44) by $\boldsymbol{d}_D^T = \boldsymbol{a}^T \boldsymbol{D}$ and eliminating $\boldsymbol{a}^T d\boldsymbol{\sigma}$ by use of (7.41) we obtain the plastic multiplier $d\lambda$ to be

$$d\lambda = \frac{1}{[A + \boldsymbol{a}^T \boldsymbol{D} \boldsymbol{a}]} \boldsymbol{a}^T \boldsymbol{d}_D d\boldsymbol{\epsilon}. \tag{7.45}$$

Or substituting (7.45) into (7.44) we obtain the complete elasto-plastic incremental stress-strain relation to be

$$d\boldsymbol{\sigma} = \boldsymbol{D}_{ep} d\boldsymbol{\epsilon}, \tag{7.46}$$

with

$$\boldsymbol{D}_{ep} = \boldsymbol{D} - \frac{\boldsymbol{d}_D \boldsymbol{d}_D^T}{A + \boldsymbol{d}_D^T \boldsymbol{a}}; \qquad \boldsymbol{d}_D = \boldsymbol{D}\boldsymbol{a}. \tag{7.47}$$

This expression for $\boldsymbol{D}_{ep}$ is similar in form to that for one dimensional application given in Page 28, Chapter 2. It now remains to determine the explicit form of the scalar term, A. The work hardening hypothesis is more general from a thermodynamic viewpoint[9] than the strain hardening hypothesis and will be employed for numerical work in this text. Therefore

$$d\kappa = \boldsymbol{\sigma}^T d\boldsymbol{\epsilon}_p. \tag{7.48}$$

Equation (7.39) can be rewritten in the form

$$F(\boldsymbol{\sigma}, \kappa) = f(\boldsymbol{\sigma}) - \sigma_Y(\kappa) = 0, \tag{7.49}$$

since the uniaxial yield stress, $\sigma_Y = \sqrt{(3)}k$. Thus from (7.43)

$$A = -\frac{1}{d\lambda}\frac{\partial F}{\partial \kappa}d\kappa = \frac{1}{d\lambda}\frac{d\sigma_Y}{d\kappa}d\kappa. \tag{7.50}$$

Note that the full differential may be employed in the last term since σ_Y is a function of κ only. Employing the normality condition in (7.48) to express $d\boldsymbol{\epsilon}_p$ we have

$$d\kappa = \boldsymbol{\sigma}^T d\boldsymbol{\epsilon}_p = \boldsymbol{\sigma}^T d\lambda \boldsymbol{a} = d\lambda \boldsymbol{a}^T \boldsymbol{\sigma}. \tag{7.51}$$

Or, for the uniaxial case $\boldsymbol{\sigma} = \bar{\sigma} = \sigma_Y$ and $d\boldsymbol{\epsilon}_p = d\bar{\epsilon}_p$ where $\bar{\sigma}$ and $\bar{\epsilon}_p$ are respectively the effective stress and strain. Thus (7.51) becomes

$$d\kappa = \sigma_Y d\bar{\epsilon}_p = d\lambda \boldsymbol{a}^T \boldsymbol{\sigma}. \tag{7.52}$$

Also, from (7.34) we have

$$\frac{d\bar{\sigma}}{d\bar{\epsilon}_p} = \frac{d\sigma_Y}{d\bar{\epsilon}_p} = H'. \tag{7.53}$$

Using Euler's theorem† applicable to all homogeneous functions of order one, we can write from (7.49)

$$\frac{\partial f}{\partial \boldsymbol{\sigma}}\boldsymbol{\sigma} = \sigma_Y. \tag{7.54}$$

Or from (7.42)

$$\boldsymbol{a}^T \boldsymbol{\sigma} = \sigma_Y. \tag{7.55}$$

Substituting (7.53) and (7.55) into (7.52) and (7.50) we obtain

$$\begin{aligned} d\lambda &= d\bar{\epsilon}_p \\ A &= H'. \end{aligned} \tag{7.56}$$

† Euler's theorem on homogeneous functions states that if $f(\mathbf{x})$ is homogeneous and of degree n then $(\partial f/\partial \mathbf{x}).\mathbf{x} = nf$.

Thus A is obtained to be the local slope of the uniaxial stress/plastic strain curve and can be determined experimentally from (7.37).

7.4 Alternative form of the yield criteria for numerical computation

For numerical computations it is convenient to rewrite the yield function in terms of alternative stress invariants. This formulation is due to Nayak[10] and its main advantage is that it permits the computer coding of the yield function and the flow rule in a general form and necessitates only the specification of three constants for any individual criterion.

The principal deviatoric stresses σ_1', σ_2', σ_3' are given as the roots of the cubic equation[11]

$$t^3 - J_2' t - J_3' = 0. \tag{7.57}$$

Noting the trigonometric identity

$$\sin^3\theta - \tfrac{3}{4}\sin\theta + \tfrac{1}{4}\sin 3\theta = 0, \tag{7.58}$$

and substituting $t = r\sin\theta$ into (7.57) we have

$$\sin^3\theta - \frac{J_2'}{r^2}\sin\theta - \frac{J_3'}{r^3} = 0. \tag{7.59}$$

Comparing (7.58) and (7.59) gives

$$r = \frac{2}{\sqrt{3}}(J_2')^{1/2}, \tag{7.60}$$

$$\sin 3\theta = -\frac{4J_3'}{r^3} = -\frac{3\sqrt{3}}{2}\,\frac{J_3'}{(J_2')^{3/2}}. \tag{7.61}$$

The first root of (7.61) with θ determined for 3θ in the range $\pm\pi/2$ is a convenient alternative to the third invariant, J_3'. By noting the cyclic nature of $\sin(3\theta + 2n\pi)$ we have immediately the three (and only three) possible values of $\sin\theta$ which define the three principal stresses. The deviatoric principal stresses are given by $t = r\sin\theta$ on substitution of the three values of $\sin\theta$ in turn. Substituting for r from (7.60) and adding the mean hydrostatic stress component gives the total principal stresses to be

$$\begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \end{Bmatrix} = \frac{2(J_2')^{\frac{1}{2}}}{\sqrt{3}} \begin{Bmatrix} \sin\left(\theta + \dfrac{2\pi}{3}\right) \\ \sin\theta \\ \sin\left(\theta + \dfrac{4\pi}{3}\right) \end{Bmatrix} + \frac{J_1}{3}\begin{Bmatrix} 1 \\ 1 \\ 1 \end{Bmatrix}, \tag{7.62}$$

with $\sigma_1 > \sigma_2 > \sigma_3$ and $-\pi/6 \leqslant \theta \leqslant \pi/6$. The term θ is essentially similar to the Lode parameter[1] Γ defined by $\Gamma = -\sqrt{(3)}\tan\theta$. The four yield criteria considered in Section 7.2.1 can now be rewritten in terms of J_1, J_2' and θ as follows.

The Tresca yield criterion

Substitute for σ_1 and σ_3 from (7.62) into (7.8) gives

$$\frac{2}{\sqrt{3}}(J_2')^{\frac{1}{2}}\left[\sin\left(\theta+\frac{2\pi}{3}\right)-\sin\left(\theta+\frac{4\pi}{3}\right)\right] = Y(\kappa),$$

or expanding we have

$$2(J_2')^{\frac{1}{2}}\cos\theta = Y(\kappa) = \sqrt{(3)}k(\kappa) = \sigma_Y(\kappa). \tag{7.63}$$

The physical interpretation of θ is evident from Fig. 7.2.

The Von Mises yield criterion

There is no change in this case since this yield function depends on J_2' only. From (7.9)

$$(J_2')^{\frac{1}{2}} = k(\kappa),$$

or

$$\sqrt{3}(J_2')^{\frac{1}{2}} = \sigma_Y(\kappa). \tag{7.64}$$

The Mohr–Coulomb yield criterion

Substituting from (7.62) for σ_1 and σ_3 into (7.16) results in

$$\frac{1}{3}J_1\sin\phi+(J_2')^{1/2}\left(\cos\theta-\frac{1}{\sqrt{3}}\sin\theta\sin\phi\right) = c\cos\phi. \tag{7.65}$$

The Drucker–Prager yield criterion

There is no change for this criterion and we can write directly from (7.17) that

$$\alpha J_1+(J_2')^{\frac{1}{2}} = k', \tag{7.66}$$

where α and k' are defined in (7.18) or (7.19).

In order to calculate the $\boldsymbol{D}_{ep}$ matrix in (7.47) we require to express the flow vector $\boldsymbol{a}$ in a form suitable for numerical computation. We can always write

$$\boldsymbol{a}^T = \frac{\partial F}{\partial \boldsymbol{\sigma}} = \frac{\partial F}{\partial J_1}\frac{\partial J_1}{\partial \boldsymbol{\sigma}}+\frac{\partial F}{\partial (J_2')^{1/2}}\frac{\partial (J_2')^{1/2}}{\partial \boldsymbol{\sigma}}+\frac{\partial F}{\partial \theta}\frac{\partial \theta}{\partial \boldsymbol{\sigma}}, \tag{7.67}$$

where

$$\boldsymbol{\sigma}^T = \{\sigma_x, \sigma_y, \sigma_z, \tau_{yz}, \tau_{zx}, \tau_{xy}\}.$$

Differentiating (7.61) we obtain

$$\frac{\partial \theta}{\partial \boldsymbol{\sigma}} = \frac{-\sqrt{3}}{2\cos 3\theta}\left[\frac{1}{(J_2')^{3/2}}\frac{\partial J_3}{\partial \boldsymbol{\sigma}}-\frac{3J_3}{(J_2')^2}\frac{\partial (J_2')^{1/2}}{\partial \boldsymbol{\sigma}}\right]. \tag{7.68}$$

Substituting this in (7.67) and using (7.61), we can then write

$$\boldsymbol{a} = C_1\boldsymbol{a}_1 + C_2\boldsymbol{a}_2 + C_3\boldsymbol{a}_3, \tag{7.69}$$

where

$$\boldsymbol{a}_1{}^T = \frac{\partial J_1}{\partial \boldsymbol{\sigma}} = \{1,\ 1,\ 1,\ 0,\ 0,\ 0\}$$

$$\boldsymbol{a}_2{}^T = \frac{\partial (J_2')^{1/2}}{\partial \boldsymbol{\sigma}} = \frac{1}{2(J_2')^{1/2}}\{\sigma_x',\ \sigma_y',\ \sigma_z',\ 2\tau_{yz},\ 2\tau_{zx},\ 2\tau_{xy}\}$$

$$\begin{aligned}\boldsymbol{a}_3{}^T = \frac{\partial J_3}{\partial \boldsymbol{\sigma}} = \Bigg\{&\left(\sigma_y'\sigma_z' - \tau_{yz}{}^2 + \frac{J_2'}{3}\right),\quad \left(\sigma_x'\sigma_z' - \tau_{xz}{}^2 + \frac{J_2'}{3}\right),\\ &\left(\sigma_x'\sigma_y' - \tau_{xy}{}^2 + \frac{J_2'}{3}\right),\quad 2(\tau_{xz}\tau_{xy} - \sigma_x'\tau_{yz}),\\ &2(\tau_{xy}\tau_{yz} - \sigma_y'\tau_{xz}),\quad 2(\tau_{yz}\tau_{xz} - \sigma_z'\tau_{xy})\Bigg\},\end{aligned} \tag{7.70}$$

and

$$\begin{aligned} C_1 &= \frac{\partial F}{\partial J_1}, \quad C_2 = \left(\frac{\partial F}{\partial (J_2')^{1/2}} - \frac{\tan 3\theta}{(J_2')^{1/2}}\frac{\partial F}{\partial \theta}\right),\\ C_3 &= \frac{-\sqrt{3}}{2\cos 3\theta}\,\frac{1}{(J_2')^{3/2}}\,\frac{\partial F}{\partial \theta}.\end{aligned} \tag{7.71}$$

Only the constants C_1, C_2 and C_3 are then necessary to define the yield surface. Thus we can achieve a simplicity of programming as only these three constants have to be varied between one yield surface and another. The constants C_i are given in Table 7.1 for the four yield criteria considered in Section 7.2.1 and other yield functions can be expressed in the same form with equal ease.

Table 7.1 Constants defining the yield surface in a form suitable for numerical analysis.

Yield Criterion	C_1	C_2	C_3
Tresca	0	$2\cos\theta(1+\tan\theta\tan 3\theta)$	$\dfrac{\sqrt{3}}{J_2'}\dfrac{\sin\theta}{\cos 3\theta}$
Von Mises	0	$\sqrt{3}$	0
Mohr–Coulomb	$\frac{1}{3}\sin\phi$	$\cos\theta[(1+\tan\theta\tan 3\theta) + \sin\phi(\tan 3\theta - \tan\theta)/\sqrt{3}]$	$\dfrac{(\sqrt{3}\sin\theta + \cos\theta\sin\phi)}{(2J_2'\cos 3\theta)}$
Drucker–Prager	α	1.0	0

7.5 Basic expressions for two dimensional problems

For two dimensional problems, the general expressions derived so far in this chapter have to be modified. Primarily the main alteration required is the deletion of the stress (and strain) components which vanish under the conditions of plane stress, plane strain or axial symmetry. We have only four non-zero stress or strain components, namely

$$\begin{aligned} \boldsymbol{\sigma}^T &= \{\sigma_x, \sigma_y, \tau_{xy}, \sigma_z\}, \quad \sigma_z = 0 \quad \text{for Plane Stress} \\ & \{\sigma_x, \sigma_y, \tau_{xy}, \sigma_z\}, \quad \epsilon_z = 0 \quad \text{Plane Strain} \\ & \{\sigma_r, \sigma_z, \tau_{rz}, \sigma_\theta\} \quad \text{Axial Symmetry.} \end{aligned} \tag{7.72}$$

From Fig. 7.8 it is seen that the z direction is taken as the coordinate independent direction for plane stress and plane strain. It is also found convenient to order the stress components as indicated in (7.72) with the stress in the coordinate independent direction being last.

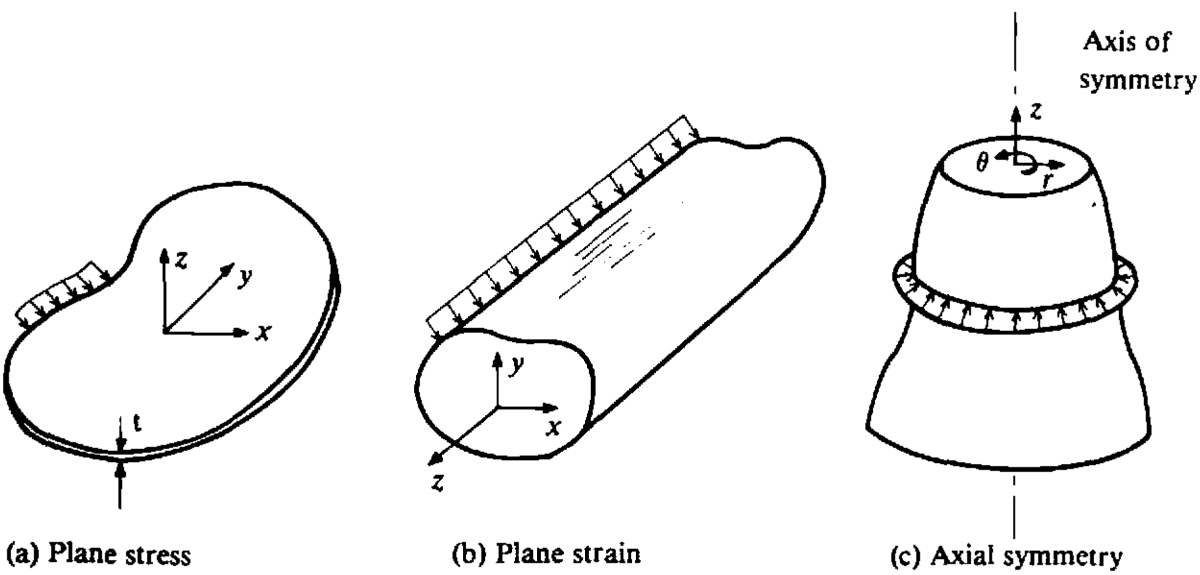


Fig. 7.8 Two-dimensional applications showing coordinate systems employed.

The explicit form of the elasticity matrix $\boldsymbol{D}$ can be written

$$\boldsymbol{D} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \left[\begin{array}{ccc|c} 1 & \dfrac{\nu}{1-\nu} & 0 & \dfrac{\nu}{1-\nu} \\ \dfrac{\nu}{1-\nu} & 1 & 0 & \dfrac{\nu}{1-\nu} \\ 0 & 0 & \dfrac{1-2\nu}{2(1-\nu)} & 0 \\ \hline \dfrac{\nu}{1-\nu} & \dfrac{\nu}{1-\nu} & 0 & 1 \end{array} \right] \quad \text{for plane strain and axial symmetry,}$$

$$\boldsymbol{D} = \frac{E}{1-\nu^2}\left[\begin{array}{ccc|c} 1 & \nu & 0 & 0 \\ \nu & 1 & 0 & 0 \\ 0 & 0 & \dfrac{1-\nu}{2} & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \quad \text{for plane stress.} \tag{7.73}$$

Note that the components corresponding to the coordinate independent direction have been included for the plane stress and strain cases. These terms will be excluded for element stiffness formulation and only the first 3×3 portion indicated will be employed. By eliminating the appropriate stress terms the expressions developed to date can be readily modified. The flow vector $\boldsymbol{a}$ becomes

$$\boldsymbol{a}^T = \left\{\frac{\partial F}{\partial \sigma_x}, \frac{\partial F}{\partial \sigma_y}, \frac{\partial F}{\partial \tau_{xy}}, \frac{\partial F}{\partial \sigma_z}\right\}, \tag{7.74}$$

with x, y and z being replaced by r, z and θ respectively for the case of axial symmetry. The specific form of the vector, $\boldsymbol{a}$ is still given by (7.69) but in this case we have from (7.70)

$$\begin{aligned} \boldsymbol{a}_1^T &= \{1,\ 1,\ 0,\ 1\} \\ \boldsymbol{a}_2^T &= \frac{1}{2(J_2')^{1/2}}\{\sigma_x',\ \sigma_y',\ 2\tau_{xy},\ \sigma_z'\} \\ \boldsymbol{a}_3^T &= \left\{\left(\sigma_y'\sigma_z' + \frac{J_2'}{3}\right),\ \left(\sigma_x'\sigma_z' + \frac{J_2'}{3}\right),\right. \\ &\qquad \left. -2\sigma_z'\tau_{xy},\ \left(\sigma_x'\sigma_y' - \tau_{xy}^2 + \frac{J_2'}{3}\right)\right\}, \end{aligned} \tag{7.75}$$

and the deviatoric stress invariants become, from (7.5)

$$\begin{aligned} J_2' &= \tfrac{1}{2}(\sigma_x'^2 + \sigma_y'^2 + \sigma_z'^2) + \tau_{xy}^2 \\ J_3' &= \sigma_z'(\sigma_z'^2 - J_2'). \end{aligned} \tag{7.76}$$

To complete the prescription of the elasto-plastic matrix $\boldsymbol{D}_{ep}$ given in (7.47) we require $\boldsymbol{d}_D$. Employing the relevant form of $\boldsymbol{D}$ from (7.73) in (7.47) results in, for plane strain and axial symmetry

$$\boldsymbol{d}_D = \begin{Bmatrix} d_1 \\ \\ d_1 \\ \\ d_3 \\ \\ d_4 \end{Bmatrix} = \begin{Bmatrix} \dfrac{E}{1+\nu}a_1+M_1 \\ \dfrac{E}{1+\nu}a_2+M_1 \\ Ga_3 \\ \dfrac{E}{1+\nu}a_4+M_1 \end{Bmatrix}, \quad M_1 = \frac{E\nu(a_1+a_2+a_4)}{(1+\nu)(1-2\nu)}, \tag{7.77}$$

where $G = E/2(1+\nu)$ is the shear modulus and $a_1 \ldots . a_4$ are the components of $\boldsymbol{a}$. For plane stress we have

$$\boldsymbol{d}_D = \begin{Bmatrix} \dfrac{E}{1+\nu}a_1+M_2 \\ \dfrac{E}{1+\nu}a_2+M_2 \\ Ga_3 \\ \dfrac{E}{1+\nu}a_4+M_2 \end{Bmatrix}, \quad M_2 = \frac{E\nu(a_1+a_2)}{1-\nu^2}. \tag{7.78}$$

7.6 Singular points on the yield surface

For many yield surfaces the flow vector $\boldsymbol{a}$ is not uniquely defined for certain stress combinations. For example this arises at the corners of the Tresca and Mohr–Coulomb criteria located by $\theta = \pm 30°$ and the direction of plastic straining there is indeterminate. Koiter[12] has provided limits within which the incremental plastic strain vector must lie. Numerical difficulties will be encountered as θ approaches $\pm 30°$ for the Tresca and Mohr–Coulomb laws since it is seen from Table 7.1 that for these values of θ both C_2 and C_3 become indeterminate. This difficulty can be overcome by returning to the original expressions (7.63) for the Tresca law and (7.65) for the Mohr–Coulomb criterion and rewriting these for the explicit values $\theta = \pm 30°$. Thus we have for the *Tresca* law

$$\sqrt{(3)}\,(J_2')^{\frac{1}{2}} = Y(\kappa) = \sqrt{(3)}k(\kappa), \tag{7.79}$$

and thus from (7.71) we have

$$C_1 = 0, \quad C_2 = \sqrt{(3)}, \quad C_3 = 0 \quad \text{for} \quad \theta = \pm 30°. \tag{7.80}$$

Physically, since (7.79) is the Von Mises criterion, this is equivalent to stating that the direction of plastic straining at the corners of the Tresca criterion is that given by the Von Mises circle which also passes through the corner (see Fig. 7.2). Similarly for the *Mohr–Coulomb* criterion we have from (7.65),

$$\frac{1}{3}J_1 \sin\phi + (J_2')^{1/2}\frac{1}{2}\left(\sqrt{3}-\frac{\sin\phi}{\sqrt{3}}\right) - c\cos\phi = 0 \quad \text{for} \quad \theta = +30^0$$

$$\frac{1}{3}J_1 \sin\phi + (J_2')^{1/2}\frac{1}{2}\left(\sqrt{3}+\frac{\sin\phi}{\sqrt{3}}\right) - c\cos\phi = 0 \qquad \theta = -30^0, \tag{7.81}$$

or from (7.71) we have

$$C_1 = \frac{1}{3}\sin\phi,\ C_2 = \frac{1}{2}\left(\sqrt{3}-\frac{\sin\phi}{\sqrt{3}}\right),\ C_3 = 0 \quad \text{for} \quad \theta = +30^0$$

$$C_1 = \frac{1}{3}\sin\phi,\ C_2 = \frac{1}{2}\left(\sqrt{3}+\frac{\sin\phi}{\sqrt{3}}\right),\ C_3 = 0 \qquad \theta = -30^0. \tag{7.82}$$

The practical approach adopted in this text is to use the general expressions for C_1, C_2, C_3 given in Table 7.1 for all values of $|\theta| \leqslant 29°$ and to then employ either (7.80) for Tresca or (7.82) for Mohr–Coulomb in the vicinity of the corners. This makes the direction of straining unique, and also satisfies the Koiter requirements. Physically this artifice corresponds to a 'rounding off' of the yield surface corners.

7.7 Finite element expressions and program structure

The basic expressions required for solution can be again obtained by use of the principle of virtual work. Consider the solid, in which the internal stresses $\boldsymbol{\sigma}$, the distributed loads/unit volume $\boldsymbol{b}$ and external applied forces $\boldsymbol{f}$ form an equilibrating field, to undergo an arbitrary virtual displacement pattern $\delta\boldsymbol{d}^*$ which result in compatible strains $\delta\boldsymbol{\epsilon}^*$ and internal displacements $\delta\boldsymbol{u}^*$. Then the principle of virtual work requires that

$$\int_\Omega (\delta\boldsymbol{\epsilon}^{*T}\boldsymbol{\sigma} - \delta\boldsymbol{u}^{*T}\boldsymbol{b})d\Omega - \delta\boldsymbol{d}^{*T}\boldsymbol{f} = 0. \tag{7.83}$$

Then the normal finite element discretising procedure leads to the following expressions for the displacements and strains within any element

$$\delta\boldsymbol{u}^* = \boldsymbol{N}\delta\boldsymbol{d}^*, \qquad \delta\boldsymbol{\epsilon}^* = \boldsymbol{B}\delta\boldsymbol{d}^*, \tag{7.84}$$

where $\boldsymbol{N}$ and $\boldsymbol{B}$ are respectively the usual matrix of shape functions and the elastic strain matrix. Then the element assembly process gives

$$\int_\Omega \delta\boldsymbol{d}^{*T}(\boldsymbol{B}^T\boldsymbol{\sigma} - \boldsymbol{N}^T\boldsymbol{b})d\Omega - \delta\boldsymbol{d}^{*T}\boldsymbol{f} = 0, \tag{7.85}$$

where the volume integration over the solid is the sum of the individual element contributions. Since this expression must hold true for any arbitrary $\delta\boldsymbol{d}^*$ value

$$\int_\Omega B^T \sigma \, d\Omega - f - \int_\Omega N^T b \, d\Omega = 0. \tag{7.86}$$

For the solution of nonlinear problems as described in Chapter 2, (7.86) will not generally be satisfied at any stage of the computation, and

$$\psi = \int_\Omega B^T \sigma \, d\Omega - \left(f + \int_\Omega N^T b \, d\Omega \right) \neq 0, \tag{7.87}$$

where ψ is the residual force vector. For an elasto-plastic situation the material stiffness is continually varying, and instantaneously the incremental stress/strain relationship is given by (7.46). For the purpose of evaluating the material tangential stiffness matrix K_T at any stage, the incremental form of (7.87) must be employed. Thus within an increment of load we have

$$\Delta\psi = \int_\Omega B^T \Delta\sigma \, d\Omega - \left(\Delta f + \int_\Omega N^T \Delta b \, d\Omega \right). \tag{7.88}$$

Substituting for $\Delta\sigma$ from (7.46) results in

$$\Delta\psi = K_T d - \left(\Delta f + \int_\Omega N^T \Delta b \, d\Omega \right), \tag{7.89}$$

where

$$K_T = \int_\Omega B^T D_{ep} B \, d\Omega. \tag{7.90}$$

Expression (7.89) is essentially identical to (2.4) and therefore the solution procedures developed in Chapter 2 can be again employed.

The programming philosophy adopted for this application follows that employed in Chapter 3 for one-dimensional elasto-plastic problems. It is suggested that the reader reviews the appropriate sections of Chapter 3 before proceeding to the remainder of this chapter. The solution techniques discussed in Chapters 2 and 3 are utilised and in particular an initial stiffness algorithm, a tangential stiffness algorithm and two options of the combined initial/tangential stiffness approach are included. An outline of the program is provided in Fig. 7.9. Many of the subroutines required are common to the corresponding linear elastic solution program and their function and structure have already been described. In particular, subroutines BMATS, CHECK1, CHECK2, DBE, ECHO, FRONT, GAUSSQ, JACOB2, LOADPS, MODPS, NODEXY and SFR2 have been described in Section 6.4. Also the standard nonlinear subroutines ALGOR, CONVER, INCREM and INPUT have been presented in Section 6.5. We will now formulate the additional subroutines required and assemble them to form a working program.

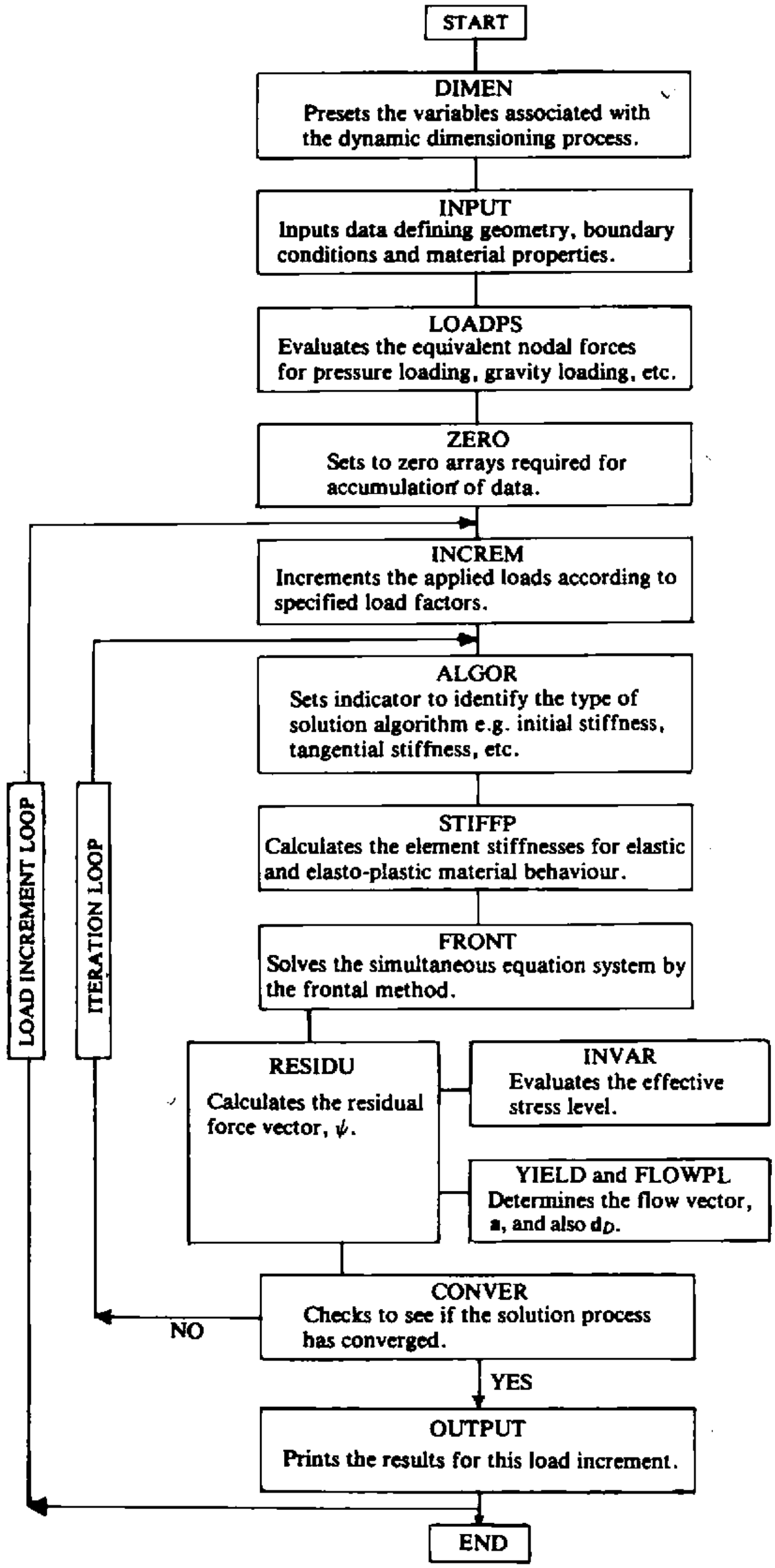


Fig. 7.9 Program organisation for two-dimensional elasto-plastic applications.

7.8 Additional program subroutines

A total of eight additional subroutines are required some of which will be common to other nonlinear applications considered in later chapters of this text.

7.8.1 Subroutine DIMEN

The function of this subroutine is to preset the values of variables employed in the program. In particular the variables associated with the dynamic dimensioning process described in Chapter 6 are defined. Thus if it is required to upgrade the magnitude of the maximum problem size which can be solved it is only necessary to modify the dimension statements in the main or master subroutine together with the variables set in subroutine DIMEN. All the variables preset in this subroutine have been previously defined and their specified values are indicated in the following listing.

```
      SUBROUTINE DIMEN(MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,MSTIF,MTOTG, DIMN
     .                 MTOTV,MVFIX,NDOFN,NPROP,NSTRE)                   DIMN
C*********************************************************************** DIMN
C                                                                       DIMN
C**** THIS SUBROUTINE PRESETS VARIABLES ASSOCIATED WITH DYNAMIC         DIMN
C     DIMENSIONING                                                      DIMN
C                                                                       DIMN
C*********************************************************************** DIMN
      MBUFA = 10                                                        DIMN
      MELEM=40                                                          DIMN
      MFRON=80                                                          DIMN
      MMATS = 5                                                         DIMN
      MPOIN=150                                                         DIMN
      MSTIF=(MFRON*MFRON-MFRON)/2.0+MFRON                               DIMN
      MTOTG = MELEM*9                                                   DIMN
      NDOFN = 2                                                         DIMN
      MTOTV = MPOIN*NDOFN                                               DIMN
      MVFIX=30                                                          DIMN
      NPROP=7                                                           DIMN
      MEVAB = NDOFN*9                                                   DIMN
      RETURN                                                            DIMN
      END                                                               DIMN
```

7.8.2 Subroutine ZERO

This subroutine merely sets to zero the contents of several arrays employed in the program. These arrays will be employed to accumulate data as the incremental and iterative process continues and they therefore require to be initialised to zero. This subroutine is self-explanatory and is presented without further comment.

```
      SUBROUTINE ZERO(ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM,  ZRO1
     .                NEVAB,NGAUS,NSTR1,NTOTG,EPSTN,EFFST,              ZRO1
     .                NTOTV,NVFIX,STRSG,TDISP,TFACT,                    ZRO1
     .                TLOAD,TREAC,MVFIX)                                ZRO1
C*********************************************************************** ZRO1
C                                                                       ZRO1
C**** THIS SUBROUTINE INITIALISES VARIOUS ARRAYS TO ZERO                ZRO1
C                                                                       ZRO1
C*********************************************************************** ZRO1
      DIMENSION ELOAD(MELEM,MEVAB),STRSG(4,MTOTG),TDISP(MTOTV),         ZRO1
     .          TLOAD(MELEM,MEVAB),TREAC(MVFIX,2),EPSTN(MTOTG),         ZRO1
     .          EFFST(MTOTG)                                            ZRO1
      TFACT=0.0                                                         ZRO1
      DO 30 IELEM=1,NELEM                                               ZRO1
      DO 30 IEVAB=1,NEVAB                                               ZRO1
      ELOAD(IELEM,IEVAB)=0.0                                            ZRO1
```

```
30 TLOAD(IELEM,IEVAB)=0.0                                  ZRO1
   DO 40 ITOTV=1,NTOTV                                     ZRO1
40 TDISP(ITOTV)=0.0                                        ZRO1
   DO 50 IVFIX=1,NVFIX                                     ZRO1
   DO 50 IDOFN=1,NDOFN                                     ZRO1
50 TREAC(IVFIX,IDOFN)=0.0                                  ZRO1
   DO 60 ITOTG=1,NTOTG                                     ZRO1
   EPSTN(ITOTG)=0.0                                        ZRO1
   EFFST(ITOTG)=0.0                                        ZRO1
   DO 60 ISTR1=1,NSTR1                                     ZRO1
60 STRSG(ISTR1,ITOTG)=0.0                                  ZRO1
   RETURN                                                  ZRO1
   END                                                     ZRO1
```

7.8.3 Subroutine INVAR

The role of this subroutine is to evaluate the various functions of stress used to indicate either initiation of or continuing plastic deformation for the four yield criteria considered in this text. More explicitly we need to calculate the items listed in Table 7.2.

Table 7.2 Effective stress and uniaxial yield stress levels for the yield criteria included in the elasto-plastic computer code.

Equation No.	Yield criterion	Stress level (effective stress)	Uniaxial (or equivalent yield stress)
(7.63)	Tresca	$2(J_2')^{1/2}\cos\theta$	σ_Y
(7.64)	Von Mises	$\sqrt{3}\,(J_2')^{1/2}$	σ_Y
(7.65)	Mohr–Coulomb	$\frac{1}{3}J_1\sin\phi+(J_2')^{1/2}$ $\times(\cos\theta-\sin\theta\sin\phi/\sqrt{3})$	$c\cos\phi$
(7.66)	Drucker–Prager	$\alpha J_1+(J_2')^{1/2}$	k'

Whether or not plastic deformation takes place at any point is governed by its stress level as monitored by the functions in the third column of Table 7.2. For plastic flow to occur this stress level must achieve the values given in the final column of Table 7.2. For the Tresca and Von Mises criteria this value is precisely the uniaxial yield stress but for the Mohr–Coulomb and Drucker–Prager criteria it is an equivalent value defined by the stress-independent terms in (7.65) and (7.66) respectively. Note that all the values given in the final column of Table 7.2 can be functions of the hardening parameter, κ.

Subroutine INVAR merely computes the effective or deviatoric stress components and then evaluates the appropriate function in the third column of Table 7.2 depending on the yield criterion being employed. The choice of yield criterion is governed by the parameter NCRIT, input in subroutine INPUT, and the available options are provided below

NCRIT = 1 Tresca yield criterion
2 Von Mises
3 Mohr–Coulomb
4 Drucker–Prager

Subroutine INVAR is now presented and descriptive notes provided.

```
      SUBROUTINE INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STEMP, INVR   1
     .                 THETA,VARJ2,YIELD)                                INVR   2
C*********************************************************************** INVR   3
C                                                                        INVR   4
C**** THIS SUBROUTINE EVALUATES THE STRESS INVARIANTS AND THE CURRENT    INVR   5
C     VALUE OF THE YIELD FUNCTION                                        INVR   6
C                                                                        INVR   7
C*********************************************************************** INVR   8
      DIMENSION DEVIA(4),PROPS(MMATS,7),STEMP(4)                        INVR   9
      ROOT3=1.73205080757                                               INVR  10
      SMEAN=(STEMP(1)+STEMP(2)+STEMP(4))/3.0                            INVR  11
      DEVIA(1)=STEMP(1)-SMEAN                                           INVR  12
      DEVIA(2)=STEMP(2)-SMEAN                                           INVR  13
      DEVIA(3)=STEMP(3)                                                 INVR  14
      DEVIA(4)=STEMP(4)-SMEAN                                           INVR  15
      VARJ2=DEVIA(3)*DEVIA(3)+0.5*(DEVIA(1)*DEVIA(1)+DEVIA(2)*DEVIA(2)  INVR  16
     . +DEVIA(4)*DEVIA(4))                                               INVR  17
      VARJ3=DEVIA(4)*(DEVIA(4)*DEVIA(4)-VARJ2)                          INVR  18
      STEFF=SQRT(VARJ2)                                                 INVR  19
      IF(STEFF.EQ.0.0) GO TO 10                                         INVR  20
      SINT3=-3.0*ROOT3*VARJ3/(2.0*VARJ2*STEFF)                          INVR  21
      IF(SINT3.GT.1.0) SINT3=1.0                                        INVR  22
      GO TO 20                                                          INVR  23
   10 SINT3=0.0                                                         INVR  24
   20 CONTINUE                                                          INVR  25
      IF(SINT3.LT.-1.0) SINT3=-1.0                                      INVR  26
      IF(SINT3.GT.1.0) SINT3=1.0                                        INVR  27
      THETA=ASIN(SINT3)/3.0                                             INVR  28
      GO TO (1,2,3,4) NCRIT                                             INVR  29
C*** TRESCA                                                             INVR  30
    1 YIELD=2.0*COS(THETA)*STEFF                                        INVR  31
      RETURN                                                            INVR  32
C*** VON MISES                                                          INVR  33
    2 YIELD=ROOT3*STEFF                                                 INVR  34
      RETURN                                                            INVR  35
C*** MOHR-COULOMB                                                       INVR  36
    3 PHIRA=PROPS(LPROP,7)*0.017453292                                  INVR  37
      SNPHI=SIN(PHIRA)                                                  INVR  38
      YIELD=SMEAN*SNPHI+STEFF*(COS(THETA)-SIN(THETA)*SNPHI/ROOT3)       INVR  39
      RETURN                                                            INVR  40
C*** DRUCKER-PRAGER                                                     INVR  41
    4 PHIRA=PROPS(LPROP,7)*0.017453292                                  INVR  42
      SNPHI=SIN(PHIRA)                                                  INVR  43
      YIELD=6.0*SMEAN*SNPHI/(ROOT3*(3.0-SNPHI))+STEFF                   INVR  44
      RETURN                                                            INVR  45
      END                                                               INVR  46
```

INVR 11–15 Compute the deviatoric stresses according to (7.7) with the order of the components being as indicated in (7.72).

INVR 16–17 Calculate the second deviatoric stress invariant, J_2'.

INVR 18 Calculate the third deviatoric stress invariant, J_3'.

INVR 19	Compute, $(J_2')^{\frac{1}{2}}$.
INVR 20–26	Evaluate $\sin 3\theta$ according to (7.61).
INVR 27	Then compute, θ. Note that the principal value is obtained as required in Section 7.4.
INVR 28	Branch according to the yield criterion being employed.
INVR 30	Evaluate the yield function in Column 3, Table 7.2 for the Tresca criterion.
INVR 33	Evaluate the yield function in Column 3, Table 7.2 for the Von Mises criterion.
INVR 36–38	Evaluate the yield function in Column 3, Table 7.2 for the Mohr–Coulomb criterion.
INVR 41–43	Evaluate the yield function in Column 3, Table 7.2 for the Drucker–Prager criterion.

7.8.4.1 Subroutine YIELDF

The function of this subroutine is to determine the flow vector $\boldsymbol{a}$ defined in (7.74). Vector $\boldsymbol{a}$ is given by (7.69) where C_1, C_2 and C_3 are given in Table 7.1 for the various yield criteria considered and the vectors $\boldsymbol{a}_1$, $\boldsymbol{a}_2$ and $\boldsymbol{a}_3$ are given by (7.75) for two dimensional applications. For the Tresca and Mohr–Coulomb yield surfaces which have singular points at $\theta = \pm 30°$ the alternative values of C_1, C_2 and C_3 given respectively in (7.80) and (7.82) must be employed.

Subroutine YIELDF is now presented and described.

```
      SUBROUTINE YIELDF(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .                  PROPS,SINT3,STEFF,THETA,VARJ2)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE FLOW VECTOR
C
C***********************************************************************
      DIMENSION AVECT(4),DEVIA(4),PROPS(MMATS,7),
     .          VECA1(4),VECA2(4),VECA3(4)
      IF(STEFF.EQ.0.0) RETURN
      FRICT=PROPS(LPROP,7)
      TANTH=TAN(THETA)
      TANT3=TAN(3.0*THETA)
      SINTH=SIN(THETA)
      COSTH=COS(THETA)
      COST3=COS(3.0*THETA)
      ROOT3=1.73205080757
C
C*** CALCULATE VECTOR A1
C
      VECA1(1)=1.0
      VECA1(2)=1.0
      VECA1(3)=0.0
      VECA1(4)=1.0
C
C*** CALCULATE VECTOR A2
C
      DO 10 ISTR1=1,NSTR1
   10 VECA2(ISTR1)=DEVIA(ISTR1)/(2.0*STEFF)
      VECA2(3)=DEVIA(3)/STEFF
```

```
C                                                                 YLDF  31
C*** CALCULATE VECTOR A3                                          YLDF  32
C                                                                 YLDF  33
      VECA3(1)=DEVIA(2)*DEVIA(4)+VARJ2/3.0                        YLDF  34
      VECA3(2)=DEVIA(1)*DEVIA(4)+VARJ2/3.0                        YLDF  35
      VECA3(3)=-2.0*DEVIA(3)*DEVIA(4)                             YLDF  36
      VECA3(4)=DEVIA(1)*DEVIA(2)-DEVIA(3)*DEVIA(3)+VARJ2/3.0      YLDF  37
      GO TO (1,2,3,4) NCRIT                                       YLDF  38
C                                                                 YLDF  39
C*** TRESCA                                                       YLDF  40
C                                                                 YLDF  41
    1 CONS1=0.0                                                   YLDF  42
      ABTHE=ABS(THETA*57.29577951308)                             YLDF  43
      IF(ABTHE.LT.29.0) GO TO 20                                  YLDF  44
      CONS2=ROOT3                                                 YLDF  45
      CONS3=0.0                                                   YLDF  46
      GO TO 40                                                    YLDF  47
   20 CONS2=2.0*(COSTH+SINTH*TANT3)                               YLDF  48
      CONS3=ROOT3*SINTH/(VARJ2*COST3)                             YLDF  49
      GO TO 40                                                    YLDF  50
C                                                                 YLDF  51
C*** VON MISES                                                    YLDF  52
C                                                                 YLDF  53
    2 CONS1=0.0                                                   YLDF  54
      CONS2=ROOT3                                                 YLDF  55
      CONS3=0.0                                                   YLDF  56
      GO TO 40                                                    YLDF  57
C                                                                 YLDF  58
C*** MOHR-COULOMB                                                 YLDF  59
C                                                                 YLDF  60
    3 CONS1=SIN(FRICT*0.017453292)/3.0                            YLDF  61
      ABTHE=ABS(THETA*57.29577951308)                             YLDF  62
      IF(ABTHE.LT.29.0) GO TO 30                                  YLDF  63
      CONS3=0.0                                                   YLDF  64
      PLUMI=1.0                                                   YLDF  65
      IF(THETA.GT.0.0) PLUMI=-1.0                                 YLDF  66
      CONS2=0.5*(ROOT3+PLUMI*CONS1*ROOT3)                         YLDF  67
      GO TO 40                                                    YLDF  68
   30 CONS2=COSTH*((1.0+TANTH*TANT3)+CONS1*(TANT3-TANTH)*ROOT3)   YLDF  69
      CONS3=(ROOT3*SINTH+3.0*CONS1*COSTH)/(2.0*VARJ2*COST3)       YLDF  70
      GO TO 40                                                    YLDF  71
C                                                                 YLDF  72
C*** DRUCKER-PRAGER                                               YLDF  73
C                                                                 YLDF  74
    4 SNPHI=SIN(FRICT*0.017453292)                                YLDF  75
      CONS1=2.0*SNPHI/(ROOT3*(3.0-SNPHI))                         YLDF  76
      CONS2=1.0                                                   YLDF  77
      CONS3=0.0                                                   YLDF  78
   40 CONTINUE                                                    YLDF  79
      DO 50 ISTR1=1,NSTR1                                         YLDF  80
   50 AVECT(ISTR1)=CONS1*VECA1(ISTR1)+CONS2*VECA2(ISTR1)+CONS3*   YLDF  81
     . VECA3(ISTR1)                                               YLDF  82
      RETURN                                                      YLDF  83
      END                                                         YLDF  84
```

YLDF 10 For the (unlikely) case of a Gauss point with zero stress (identified by $J_2' = J_3' = 0$) avoid evaluation of the flow vector.

YLDF 11 Identify FRICT as the friction angle ϕ for Mohr–Coulomb and Drucker–Prager materials.

YLDF 12–13 Evaluate $\tan\theta$ and $\tan 3\theta$.

YLDF 14–16 Evaluate $\sin\theta$, $\cos\theta$ and $\cos 3\theta$.

YLDF 17 Compute $\sqrt{(3)}$.

YLDF 21–24 Evaluate $\boldsymbol{a}_1$ according to (7.75).

YLDF 28–30 Evaluate $\boldsymbol{a}_2$ according to (7.75). Note that STEFF and DEVIA are transferred via the argument list from subroutine INVAR.

YLDF 34–37 Evaluate $\boldsymbol{a}_3$ according to (7.75).

YLDF 38 Branch according to the yield criterion being employed.

YLDF 41–49 Compute the constants C_1, C_2 and C_3 for a Tresca material according to Table 7.1. In the vicinity of a singular point, identified by $|\theta|>29.0°$ evaluate C_1, C_2 and C_3 according to (7.80).

YLDF 53–55 Compute C_1, C_2 and C_3 for a Von Mises material according to Table 7.1.

YLDF 61–67 Compute C_1, C_2 and C_3 for the Mohr–Coulomb criterion. In the vicinity of a singular point defined by $|\theta|>29.0°$ evaluate C_1, C_2 and C_3 according to (7.82).

YLDF 75–78 Calculate C_1, C_2 and C_3 for the Drucker–Prager yield criterion.

YLDF 80–82 Evaluate $\boldsymbol{a}$ according to (7.69).

7.8.4.2 Subroutine FLOWPL

The main purpose of this subroutine is to determine the vector $\boldsymbol{d}_D$ according to either (7.77) or (7.78) depending on the type of analysis being undertaken. In the program presented in this chapter only a linear form of strain hardening is explicitly considered, with the coding of alternative models being left as an exercise for the reader. In this case the term H' in (7.37) becomes a constant and is specified as a material property.

Subroutine FLOWPL is now listed and described.

```
      SUBROUTINE FLOWPL(AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS)FLPL   1
C*********************************************************************** FLPL   2
C                                                                        FLPL   3
C**** THIS SUBROUTINE EVALUATES THE PLASTIC D VECTOR                     FLPL   4
C                                                                        FLPL   5
C*********************************************************************** FLPL   6
      DIMENSION AVECT(4),DVECT(4),PROPS(MMATS,7)                         FLPL   7
      YOUNG=PROPS(LPROP,1)                                               FLPL   8
      POISS=PROPS(LPROP,2)                                               FLPL   9
      HARDS=PROPS(LPROP,6)                                               FLPL  10
      FMUL1=YOUNG/(1.0+POISS)                                            FLPL  11
      IF(NTYPE.EQ.1) GO TO 60                                            FLPL  12
      FMUL2=YOUNG*POISS*(AVECT(1)+AVECT(2)+AVECT(4))/((1.0+POISS)*       FLPL  13
     . (1.0-2.0*POISS))                                                  FLPL  14
      DVECT(1)=FMUL1*AVECT(1)+FMUL2                                      FLPL  15
      DVECT(2)=FMUL1*AVECT(2)+FMUL2                                      FLPL  16
      DVECT(3)=0.5*AVECT(3)*YOUNG/(1.0+POISS)                            FLPL  17
      DVECT(4)=FMUL1*AVECT(4)+FMUL2                                      FLPL  18
      GO TO 70                                                           FLPL  19
```

```
   60 FMUL3=YOUNG*POISS*(AVECT(1)+AVECT(2))/(1.0-POISS*POISS)          FLPL 20
      DVECT(1)=FMUL1*AVECT(1)+FMUL3                                    FLPL 21
      DVECT(2)=FMUL1*AVECT(2)+FMUL3                                    FLPL 22
      DVECT(3)=0.5*AVECT(3)*YOUNG/(1.0+POISS)                          FLPL 23
      DVECT(4)=FMUL1*AVECT(4)+FMUL3                                    FLPL 24
   70 DENOM=HARDS                                                      FLPL 25
      DO 80 ISTR1=1,NSTR1                                              FLPL 26
   80 DENOM=DENOM+AVECT(ISTR1)*DVECT(ISTR1)                            FLPL 27
      ABETA=1.0/DENOM                                                  FLPL 28
      RETURN                                                           FLPL 29
      END                                                              FLPL 30
```

FLPL 8	Identify YOUNG as the elastic modulus, E.
FLPL 9	Identify POISS as the Poisson's ratio, ν.
FLPL 10	Identify HARDS as H' for linear strain hardening.
FLPL 13–18	Evaluate $\boldsymbol{d}_D$ according to (7.77) for plane strain and axisymmetric situations.
FLPL 20–24	Evaluate $\boldsymbol{d}_D$ according to (7.78) for plane stress problems.
FLPL 26–28	Compute $1/(H'+\boldsymbol{d}_D{}^T\boldsymbol{a})$ for later evaluation of the elasto-plastic matrix $\boldsymbol{D}_{ep}$ in (7.47).

7.8.5 Subroutine STIFFP

This subroutine evaluates the stiffness matrix for each element in turn and differs from the linear elastic version, described in Section 6.3.2, only in that the elasticity matrix $\boldsymbol{D}$ is replaced (for the tangential stiffness approach at least) by the elasto-plastic matrix $\boldsymbol{D}_{ep}$ defined in (7.47). This subroutine is called only when the element stiffnesses are to be reformulated as controlled by variable KRESL defined in subroutine ALGOR. Obviously the element stiffnesses must be calculated for the first iteration of the first load increment and elastic behaviour must be assumed. Every other time this subroutine is accessed the stiffnesses are to be recalculated to account for any plastic deformation of the material and consequently the $\boldsymbol{D}_{ep}$ matrix must be employed. Apart from this change the element stiffness formulation process is identical to that for elastic materials as described in Section 6.3.2.

Subroutine STIFFP will now be described and explanatory notes provided.

```
      SUBROUTINE STIFFP(COORD,EPSTN.IINCS,LNODS,MATNO,MEVAB,MMATS,
     .                  MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE,NSTRE,
     .                  NSTR1,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     .                  STRSG,NTYPE,NCRIT)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE STIFFNESS MATRIX FOR EACH ELEMENT
C     IN TURN
C
C***********************************************************************
      DIMENSION BMATX(4,18),CARTD(2,9),COORD(MPOIN,2),DBMAT(4,18),
     .          DERIV(2,9),DEVIA(4),DMATX(4,4),
     .          ELCOD(2,9),EPSTN(MTOTG),ESTIF(18,18),LNODS(MELEM,9),
     .          MATNO(MELEM),POSGP(4),PROPS(MMATS,7),SHAPE(9),
     .          WEIGP(4),STRES(4),STRSG(4,MTOTG),
     .          DVECT(4),AVECT(4),GPCOD(2,9)
      TWOPI=6.283185308
      REWIND 1
```

```
      KGAUS=0
C
C*** LOOP OVER EACH ELEMENT
C
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 10 INODE=1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      IPOSN=(LNODE-1)*2
      DO 10 IDIME=1,2
      IPOSN=IPOSN+1
   10 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
      THICK=PROPS(LPROP,3)
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
C
      DO 20 IEVAB=1,NEVAB
      DO 20 JEVAB=1,NEVAB
   20 ESTIF(IEVAB,JEVAB)=0.0
      KGASP=0
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      DO 50 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      DO 50 JGAUS=1,NGAUS
      ETASP=POSGP(JGAUS)
      KGASP=KGASP+1
      KGAUS=KGAUS+1
C
C*** EVALUATE THE D-MATRIX
C
      CALL MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL VOLUME,ETC.
C
      CALL        SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL        JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .                   NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(THICK.NE.0.0) DVOLU=DVOLU*THICK
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      IF(IINCS.EQ.1) GO TO 80
      IF(EPSTN(KGAUS).EQ.0.0) GO TO 80
      DO 90 ISTR1=1,NSTR1
   90 STRES(ISTR1)=STRSG(ISTR1,KGAUS)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,
     .           THETA,VARJ2,YIELD)
      CALL YIELDF(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .            PROPS.SINT3,STEFF,THETA,VARJ2)
      CALL FLOWPL(AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS)
      DO 100 ISTRE=1,NSTRE
      DO 100 JSTRE=1,NSTRE
  100 DMATX(ISTRE,JSTRE)=DMATX(ISTRE,JSTRE)-ABETA*DVECT(ISTRE)*
     . DVECT(JSTRE)
   80 CONTINUE
      CALL        DBE(BMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTRE,NSTR1)
```

```
C                                                                        STFP  83
C*** CALCULATE THE ELEMENT STIFFNESSES                                   STFP  84
C                                                                        STFP  85
      DO 30 IEVAB=1,NEVAB                                                STFP  86
      DO 30 JEVAB=IEVAB,NEVAB                                            STFP  87
      DO 30 ISTRE=1,NSTRE                                                STFP  88
   30 ESTIF(IEVAB,JEVAB)=ESTIF(IEVAB,JEVAB)+BMATX(ISTRE,IEVAB)*          STFP  89
     . DBMAT(ISTRE,JEVAB)*DVOLU                                          STFP  90
   50 CONTINUE                                                           STFP  91
C                                                                        STFP  92
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX                STFP  93
C                                                                        STFP  94
      DO 60 IEVAB=1,NEVAB                                                STFP  95
      DO 60 JEVAB=1,NEVAB                                                STFP  96
   60 ESTIF(JEVAB,IEVAB)=ESTIF(IEVAB,JEVAB)                              STFP  97
C                                                                        STFP  98
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX AND SAMPLING POINT         STFP  99
C    COORDINATES FOR EACH ELEMENT ON DISC FILE                           STFP 100
C                                                                        STFP 101
      WRITE(1) ESTIF                                                     STFP 102
   70 CONTINUE                                                           STFP 103
      RETURN                                                             STFP 104
      END                                                                STFP 105
```

STFP 17 Compute the value of 2π.

STFP 18 Rewind the disc file on which the element stiffness matrices will be stored in turn.

STFP 19 Set to zero the counter which indicates the overall Gauss point location. So KGAUS ranges from 1 to NGAUS*NGAUS*NELEM.

STFP 23 Enter the loop over each element in the structure.

STFP 24 Identify the material property type of the current element.

STFP 28–33 Store the element nodal coordinates in the local array ELCOD for convenient use later.

STFP 34 Identify the element thickness.

STFP 38–40 Zero the element stiffness array.

STFP 41 Set to zero the element Gauss point counter. So KGASP ranges from 1 to NGAUS*NGAUS.

STFP 45–48 Enter the numerical integration loops and locate the position (ξ, η) of the current point.

STFP 49–50 Increment the local and global Gauss point counters.

STFP 54 Call subroutine MODPS to evaluate the elasticity matrix, $\boldsymbol{D}$.

STFP 58 Evaluate the shape functions N_i and the derivatives $\partial N_i/\partial \xi$, $\partial N_i/\partial \eta$ for the current Gauss point.

STFP 59–60 Evaluate the Gauss point coordinates, GPCOD(IDIME, KGASP), the determinant of the Jacobian matrix, $|\boldsymbol{J}|$ and the Cartesian derivatives of the shape functions $\partial N_i/\partial x$, $\partial N_i/\partial y$ (or $\partial N_i/\partial r$, $\partial N_i/\partial z$ for axisymmetric problems).

STFP 61–63 Calculate the elemental volume for numerical integration as $|\boldsymbol{J}|\, W_\xi W_\eta$ taking care to multiply by the appropriate thickness or by $2\pi r$ for axisymmetric problems. Note that if a zero thickness is specified it is automatically taken to be unity.

STFP 67 Evaluate the $\boldsymbol{B}$ matrix.

STFP 68 For the first time avoid the replacement of $\boldsymbol{D}$ by $\boldsymbol{D}_{ep}$, as defined in (7.47).

STFP 69 Also for Gauss points at which the behaviour is elastic avoid the replacement of $\boldsymbol{D}$ by $\boldsymbol{D}_{ep}$.

STFP 70–71 Store the total current stresses in the array STRES.

STFP 72–76 Call subroutines INVAR, YIELDF and FLOWPL to evaluate the vectors $\boldsymbol{a}$, (AVECT) and $\boldsymbol{d}_D$, (DVECT) and ABETA $= 1/(H' + \boldsymbol{d}_D{}^T\boldsymbol{a})$.

STFP 77–80 Evaluate $\boldsymbol{D}_{ep}$ according to (7.47).

STFP 82 Evaluate $\boldsymbol{D}_{ep}\boldsymbol{B}$.

STFP 86–90 Compute the upper triangle of the element stiffness matrix as

$$\int_{\Omega} \boldsymbol{B}^T \boldsymbol{D}_{ep} \boldsymbol{B}\, d\Omega$$

STFP 91 End of loop for numerical integration.

STFP 95–97 Complete the lower triangle of the element stiffness matrix by symmetry.

STFP 102 Store the element stiffness matrix on disc file 1.

STFP 103 Return to process the next element.

7.8.6 Subroutine LINEAR

The purpose of this subroutine is merely to determine the stresses from given displacements assuming linear elastic behaviour. This subroutine is employed in the residual force calculation to be described in the next section. The element displacement components, ELDIS(IDOFN, INODE) are entered into the subroutine, the strain components at the Gauss point under consideration, STRAN(ISTR1) calculated and finally the stress components are evaluated and stored in STRES(ISTR1).

The subroutine is now listed and described.

```
      SUBROUTINE LINEAR(CARTD,DMATX,ELDIS,LPROP,MMATS,NDOFN,NNODE,NSTRE,
     .                  NTYPE,PROPS,STRAN,STRES,KGASP,GPCOD,SHAPE)
C**********************************************************************
C
C**** THIS SUBROUTINE EVALUATES STRESSES AND STRAINS ASSUMING LINEAR
C     ELASTIC BEHAVIOUR
C
C**********************************************************************
      DIMENSION AGASH(2,2),CARTD(2,9),DMATX(4,4),ELDIS(2,9),
     .          PROPS(MMATS,7),STRAN(4),STRES(4),
     .          GPCOD(2,9),SHAPE(9)
      POISS=PROPS(LPROP,2)
      DO 20 IDOFN=1,NDOFN
      DO 20 JDOFN=1,NDOFN
      BGASH=0.0
      DO 10 INODE=1,NNODE
```

```
   10 BGASH=BGASH+CARTD(JDOFN,INODE)*ELDIS(IDOFN,INODE)           LINR 17
   20 AGASH(IDOFN,JDOFN)=BGASH                                    LINR 18
C                                                                 LINR 19
C*** CALCULATE THE STRAINS                                        LINR 20
C                                                                 LINR 21
      STRAN(1)=AGASH(1,1)                                         LINR 22
      STRAN(2)=AGASH(2,2)                                         LINR 23
      STRAN(3)=AGASH(1,2)+AGASH(2,1)                              LINR 24
      STRAN(4)=0.0                                                LINR 25
      DO 30 INODE=1,NNODE                                         LINR 26
   30  STRAN(4)=STRAN(4)+ELDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP) LINR 27
C                                                                 LINR 28
C*** AND THE CORRESPONDING STRESSES                               LINR 29
C                                                                 LINR 30
      DO 40 ISTRE=1,NSTRE                                         LINR 31
      STRES(ISTRE)=0.0                                            LINR 32
      DO 40 JSTRE=1,NSTRE                                         LINR 33
   40 STRES(ISTRE)=STRES(ISTRE)+DMATX(ISTRE,JSTRE)*STRAN(JSTRE)   LINR 34
      IF(NTYPE.EQ.1) STRES(4)=0.0                                 LINR 35
      IF(NTYPE.EQ.2) STRES(4)=POISS*(STRES(1)+STRES(2))           LINR 36
      RETURN                                                      LINR 37
      END                                                         LINR 38
```

LINR 12 Identify POISS as the Poisson's ratio of the element material.

LINR 13–18 Calculate the Cartesian derivatives of the Gauss point displacement components $\partial u/\partial x$, $\partial u/\partial y$, $\partial v/\partial x$, $\partial v/\partial y$.

LINR 22–27 Evaluate the strain components at the Gauss point according to

$$\boldsymbol{\epsilon} = \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \\ \epsilon_z \cdot \end{Bmatrix} = \begin{Bmatrix} \dfrac{\partial u}{\partial x} \\ \dfrac{\partial v}{\partial y} \\ \dfrac{\partial u}{\partial y}+\dfrac{\partial v}{\partial x} \\ 0 \end{Bmatrix} \text{for plane problems,}$$

$$\boldsymbol{\epsilon} = \begin{Bmatrix} \epsilon_r \\ \epsilon_z \\ \gamma_{rz} \\ \epsilon_\theta \end{Bmatrix} = \begin{Bmatrix} \dfrac{\partial u}{\partial r} \\ \dfrac{\partial w}{\partial z} \\ \dfrac{\partial u}{\partial z}+\dfrac{\partial w}{\partial r} \\ \dfrac{u}{r} \end{Bmatrix} \text{for axisymmetric problems.}$$

LINR 31–34 Calculate the stress components, assuming elastic behaviour, according to $\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\epsilon}$.

LINR 35–36 For a plane stress problem set $\sigma_z = 0$ and set $\sigma_z = \nu(\sigma_x + \sigma_y)$ for plane strain situations.

7.8.7 Subroutine RESIDU

The function of this subroutine is to evaluate the nodal forces which are statically equivalent to the stress field satisfying elasto-plastic conditions. Comparison of these equivalent nodal forces with the applied loads gives the residual forces, according to (2.4), and this operation is carried out in subroutine CONVER. Therefore RESIDU performs the same task for two-dimensional continua as subroutine REFOR3 undertook for uniaxial situations, and the reader is urged to review Section 3.12.2 before proceeding further. The logic applied in this subroutine is almost identical to that applied in Section 3.12.2. Below we reproduce the essential steps in an abbreviated form and expand only the steps which pertain to the case of two dimensional solids.

During the application of an increment of load an element, or part of an element, may yield. All stress and strain quantities are monitored at each Gaussian integration point and therefore we can determine whether or not plastic deformation has occurred at such points. Consequently an element can behave partly elastically and partly elasto-plastically if some, but not all, Gauss points indicate plastic yielding. For any load increment it is necessary to determine what proportion is elastic and which part produces plastic deformation and then adjust the stress and strain terms until the yield criterion and the constitutive laws are satisfied. The procedure adopted is as follows.

Step a. The applied loads for the r^{th} iteration are the residual forces ψ^{r-1}, given by (2.4) which give rise to displacement increments $d\boldsymbol{d}^r$, according to (2.12), and strain increments $d\boldsymbol{\epsilon}^r$.

$\Delta\Psi^r = -[H^r$ [illegible] $]^{-1}\psi(\sigma^r)$ [illegible]

Step b. Compute the incremental stress changes, $d\boldsymbol{\sigma}_e{}^r$ as $d\boldsymbol{\sigma}_e{}^r = \boldsymbol{D}d\boldsymbol{\epsilon}^r$ where the subscript e denotes that we are assuming elastic behaviour.

Step c. Accumulate the total stress for each element Gauss point as $\boldsymbol{\sigma}_e{}^r = \boldsymbol{\sigma}^{r-1} + d\boldsymbol{\sigma}_e{}^r$ where $\boldsymbol{\sigma}^{r-1}$ are the converged stresses for iteration $r-1$.

Step d. The next step depends on whether or not yielding took place at the Gauss point during the $(r-1)^{\text{th}}$ iteration. Therefore we check if $\bar{\sigma}^{r-1} > \sigma_Y = \sigma_Y{}^\circ + H'\bar{\epsilon}_p{}^{r-1}$, where $\bar{\sigma}^{r-1}$ is the effective stress given by Column 3, Table 7.2, σ_Y is the uniaxial yield stress, (Column 4, Table 7.2), H' is the linear strain hardening parameter and $\bar{\epsilon}_p{}^{r-1}$ is the effective plastic strain existing at the end of the $(r-1)^{\text{th}}$ iteration. This expression is identical to the uniaxial case, Section 3.12.2, with all quantities replaced by the effective or equivalent values. If the answer is:

<table>
<tr><th colspan="2">YES</th><th colspan="2">NO</th></tr>
<tr><td colspan="2">The Gauss point had previously yielded. Now check to see if $\bar{\sigma}_e{}^r > \bar{\sigma}^{r-1}$ where $\bar{\sigma}_e{}^r$ is the effective stress, Col. 3, Table 7.2 based on stresses $\boldsymbol{\sigma}_e{}^r$. If the answer is:</td><td colspan="2">Which implies that the Gauss point had not previously yielded. Now check to see if $\bar{\sigma}_e{}^r > \sigma_Y{}^0$. If the answer is:</td></tr>
<tr><th>NO</th><th>YES</th><th>NO</th><th>YES</th></tr>
<tr><td>The Gauss point is unloading elastically and therefore go directly to Step g.</td><td>The Gauss point had yielded previously and the stress is still increasing. Therefore all the excess stress $\boldsymbol{\sigma}_e{}^r - \boldsymbol{\sigma}^{r-1}$ must be reduced to the yield surface as indicated in Fig. 7.10(a). Therefore the factor R which defines the portion of stress which must be modified to satisfy the yield criterion is equal to 1.</td><td>The Gauss point is still elastic and therefore go directly to Step g.</td><td>The Gauss point has yielded during application of load corresponding to this iteration as shown in Fig. 7.10(b). The portion of the stress greater than the yield value must be reduced to the yield surface. The reduction factor R is given from Fig. 7.10(b) to be
$$R = \frac{AB}{AC} = \frac{\bar{\sigma}_e{}^r - \sigma_Y}{\bar{\sigma}_e{}^r - \bar{\sigma}^{r-1}}.$$</td></tr>
</table>

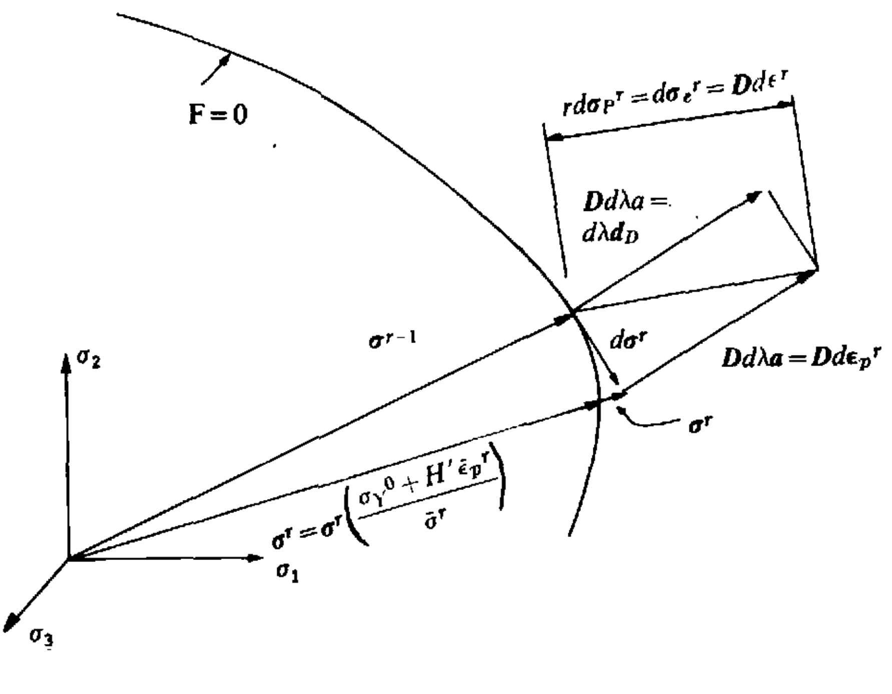


Fig. 7.10(a) Incremental stress changes in an already yielded point in an elasto-plastic continuum.

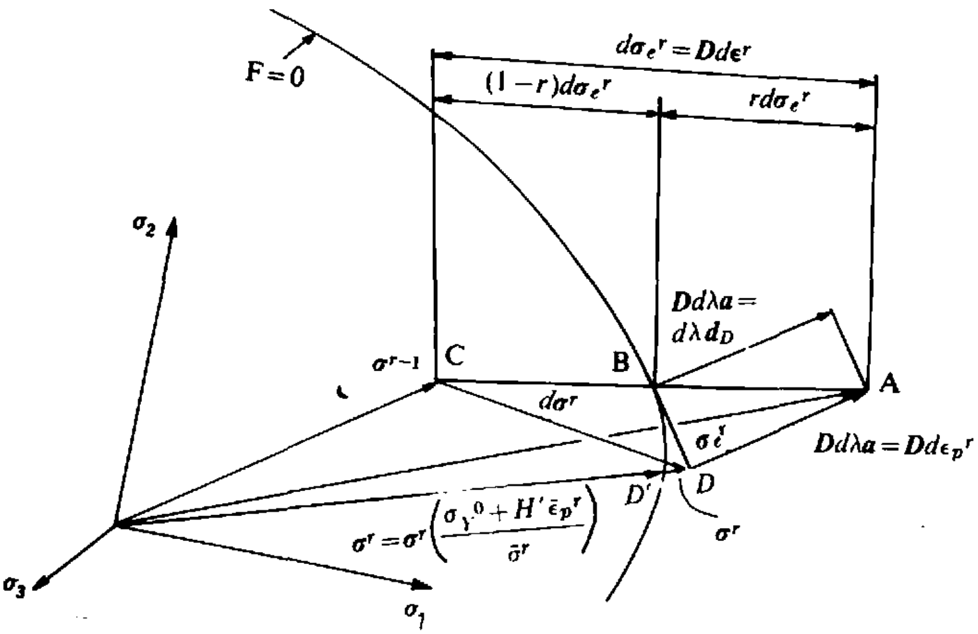


Fig. 7.10(b) Incremental stress changes at a point in an elasto-plastic continuum at initial yield.

Step e. *For yielded Gauss points only* compute the portion of the total stress which satisfies the yield criterion as $\boldsymbol{\sigma}^{r-1}+(1-R)d\boldsymbol{\sigma}_e{}^r$.

Step f. The remaining portion of stress, $Rd\boldsymbol{\sigma}_e{}^r$ must be effectively eliminated in some way. The point A must be brought onto the yield surface by allowing plastic deformation to occur. Physically this can be described as follows. On loading from point C, the stress point moves elastically until the yield surface is met at B. Elastic behaviour beyond this point would result in a final stress state defined by point A. However in order to satisfy the yield criterion, the stress point cannot move outside the yield surface and consequently the stress point can only traverse the surface until both equilibrium conditions and the constitutive relation are satisfied. From (7.45), (7.46) and (7.47) we have

$$d\boldsymbol{\sigma}^r = \boldsymbol{D}d\boldsymbol{\epsilon}^r - d\lambda \boldsymbol{d}_D, \tag{7.91}$$

or

$$\boldsymbol{\sigma}^r = \boldsymbol{\sigma}^{r-1} + d\boldsymbol{\sigma}_e{}^r - d\lambda \boldsymbol{d}_D, \tag{7.92}$$

which gives the total stresses $\boldsymbol{\sigma}^r$ satisfying elasto-plastic conditions when the stresses are incremented from $\boldsymbol{\sigma}^{r-1}$. Expression (7.92) is illustrated vectorially in Fig. 7.10 and the reader should note the similarity to Fig. 3.7(a). It is seen that if a finite sized stress increment is taken, the final stress point D, corresponding to $\boldsymbol{\sigma}^r$, may depart from the yield surface. This discrepancy can be practically eliminated by ensuring that the load increments considered in solution are sufficiently small. However the point D can be reduced to the yield surface by simply scaling the vector $\boldsymbol{\sigma}^r$. Denoting the effective stress, given by Col. 3, Table 7.2, due to stress $\boldsymbol{\sigma}^r$ as $\bar{\sigma}^r$ and noting that this value should coincide with $\sigma_Y = \sigma_Y{}^0 + H'\bar{\epsilon}_p{}^r$ if the point D lies on the yield surface, the appropriate scaling factor is readily seen to be

$$\boldsymbol{\sigma}^r = \boldsymbol{\sigma}^r\left(\frac{\sigma_Y{}^0 + H'\,\bar{\epsilon}_p{}^r}{\bar{\sigma}^r}\right). \tag{7.93}$$

This represents a scaling of the vector $\boldsymbol{\sigma}^r$ which implies that the individual stress components are proportionally reduced. The normality condition for the plastic strain increment is evident from Fig. 7.10 since $\boldsymbol{D}d\lambda\boldsymbol{a} = \boldsymbol{D}d\boldsymbol{\varepsilon}_p$.

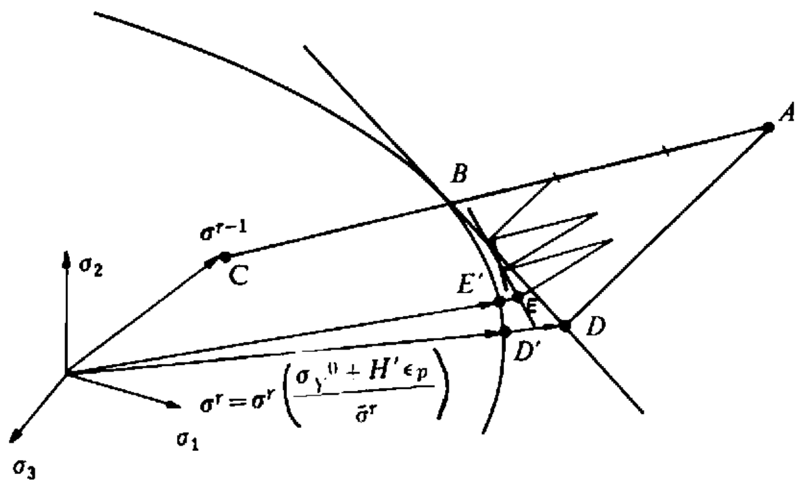


Fig. 7.11 Refined process for reducing a stress point to the yield surface.

If relatively large load increment sizes are to be permitted the process described above can lead to an inaccurate prediction of the final point D on the yield surface if the stress point is in the vicinity of a region of large curvature of the yield surface. This is illustrated in Fig. 7.11 where the process of reducing the elastic stress to the yield surface is shown to end in the stress point D which is then scaled down to the yield surface to give point D'. Greater accuracy can be achieved by relaxing the excess stress to the yield surface in several stages.* Fig. 7.11 shows the case where the excess stress is divided into three equal parts and each increment reduced to the yield surface in turn. After the three reduction cycles to the stress point E the drift away from the yield surface can be corrected by simple scaling to give the final stress point E'. It is seen that the final

* Alternative procedures for this operation are presented in Refs. 18 and 19 whilst a completely different approach to stress projection is followed in Ref. 20.

points D' and E' can be significantly different. An additional refinement which can be introduced is to scale the stress point to the yield surface after the reduction process for each cycle and not only after the final cycle as shown in Fig. 7.11. Obviously the greater the number of steps into which the excess stress AB is divided, the greater the accuracy. However the computation for each step is relatively expensive since the vectors $\boldsymbol{a}$ and $\boldsymbol{d}_D$ have to be calculated at each stage. Clearly a balance must be sought and in this text the following criterion is adopted. The excess stress $Rd\boldsymbol{\sigma}_e{}^r$ is divided into m parts where m is given by the nearest integer which is less than

$$\left(\frac{\bar{\sigma}_e{}^r - \sigma_Y}{\sigma_Y{}^0}\right)8 + 1, \tag{7.94}$$

where $\bar{\sigma}_e{}^r - \sigma_Y$ gives a measure of the excess stress AB and $\sigma_Y{}^\circ$ is the initial uniaxial yield stress in Col. 4, Table 7.2 before the onset of work hardening. This criterion can be readily amended by the user.

Step g. *For elastic Gauss points only* calculate $\boldsymbol{\sigma}^r$ as $\boldsymbol{\sigma}^r = \boldsymbol{\sigma}^{r-1} + d\boldsymbol{\sigma}_e{}^r$.

Step h. Finally, calculate the equivalent nodal forces from the element stresses according to

$$(\boldsymbol{f}^{(e)})^r = \int_{\Omega^{(e)}} \boldsymbol{B}^T \boldsymbol{\sigma}^r \, d\Omega. \tag{7.95}$$

Subroutine RESIDU is now listed and described.

```
      SUBROUTINE RESIDU(ASDIS,COORD,EFFST,ELOAD,FACTO,IITER,LNODS,
     .                  LPROP,MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,NDOFN,
     .                  NELEM,NEVAB,NGAUS,NNODE,NSTR1,NTYPE,POSGP,PROPS,
     .                  NSTRE,NCRIT,STRSG,WEIGP,TDISP,EPSTN)
C*********************************************************************
C
C**** THIS SUBROUTINE REDUCES THE STRESSES TO THE YIELD SURFACE AND
C     EVALUATES THE EQUIVALENT NODAL FORCES
C
C*********************************************************************
      DIMENSION ASDIS(MTOTV),AVECT(4),CARTD(2,9),COORD(MPOIN,2),
     .          DEVIA(4),DVECT(4),EFFST(MTOTG),ELCOD(2,9),ELDIS(2,9),
     .          ELOAD(MELEM,18),LNODS(MELEM,9),POSGP(4),PROPS(MMATS,7),
     .          STRAN(4),STRES(4),STRSG(4,MTOTG),
     .          WEIGP(4),DLCOD(2,9),DESIG(4),SIGMA(4),SGTOT(4),
     .          DMATX(4,4),DERIV(2,9),SHAPE(9),GPCOD(2,9),
     .          EPSTN(MTOTG),TDISP(MTOTV),MATNO(MELEM),BMATX(4,18)
      ROOT3=1.73205080757
      TWOPI=6.283185308
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=0.0
      KGAUS=0
      DO 20 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      UNIAX=PROPS(LPROP,5)
      HARDS=PROPS(LPROP,6)
```

```
      FRICT=PROPS(LPROP,7)
      IF(NCRIT.EQ.3) UNIAX=PROPS(LPROP,5)*COS(FRICT*0.017453292)
      IF(NCRIT.EQ.4) UNIAX=6.0*PROPS(LPROP,5)*COS(FRICT*0.017453292)/
     . (ROOT3*(3.0-SIN(FRICT*0.017453292)))
C
C*** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 30 INODE =1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      NPOSN=(LNODE-1)*NDOFN
      DO 30 IDOFN=1,NDOFN
      NPOSN=NPOSN+1
      ELCOD(IDOFN,INODE)=COORD(LNODE,IDOFN)
   30 ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
      CALL      MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
      THICK=PROPS(LPROP,3)
      KGASP=0
      DO 40 IGAUS=1,NGAUS
      DO 40 JGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      KGAUS=KGAUS+1
      KGASP=KGASP+1
      CALL      SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL      JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .                 NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(THICK.NE.0.0) DVOLU=DVOLU*THICK
      CALL BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      CALL LINEAR(CARTD,DMATX,ELDIS,LPROP,MMATS,NDOFN,NNODE,NSTRE,
     .            NTYPE,PROPS,STRAN,STRES,KGASP,GPCOD,SHAPE)
      PREYS=UNIAX+EPSTN(KGAUS)*HARDS
      DO 150 ISTR1=1,NSTR1
      DESIG(ISTR1)=STRES(ISTR1)
  150 SIGMA(ISTR1)=STRSG(ISTR1,KGAUS)+STRES(ISTR1)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SIGMA,
     .           THETA,VARJ2,YIELD)
      ESPRE=EFFST(KGAUS)-PREYS
      IF(ESPRE.GE.0.0) GO TO 50
      ESCUR=YIELD-PREYS
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=ESCUR/(YIELD-EFFST(KGAUS))
      GO TO 70
   50 ESCUR=YIELD-EFFST(KGAUS)
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=1.0
   70 MSTEP=ESCUR*8.0/UNIAX+1.0
      ASTEP=MSTEP
      REDUC=1.0-RFACT
      DO 80 ISTR1=1,NSTR1
      SGTOT(ISTR1)=STRSG(ISTR1,KGAUS)+REDUC*STRES(ISTR1)
   80 STRES(ISTR1)=RFACT*STRES(ISTR1)/ASTEP
      DO 90 ISTEP=1,MSTEP
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SGTOT,
     .           THETA,VARJ2,YIELD)
      CALL YIELDF(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .            PROPS,SINT3,STEFF,THETA,VARJ2)
      CALL FLOWPL(AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS)
      AGASH=0.0
      DO 100 ISTR1=1,NSTR1
  100 AGASH=AGASH+AVECT(ISTR1)*STRES(ISTR1)
      DLAMD=AGASH*ABETA
```

```
      IF(DLAMD.LT.0.0) DLAMD=0.0                                        RSDU  92
      BGASH=0.0                                                         RSDU  93
      DO 110 ISTR1=1,NSTR1                                              RSDU  94
      BGASH=BGASH+AVECT(ISTR1)*SGTOT(ISTR1)                             RSDU  95
  110 SGTOT(ISTR1)=SGTOT(ISTR1)+STRES(ISTR1)-DLAMD*DVECT(ISTR1)         RSDU  96
      EPSTN(KGAUS)=EPSTN(KGAUS)+DLAMD*BGASH/YIELD                       RSDU  97
   90 CONTINUE                                                          RSDU  98
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SGTOT,       RSDU  99
     .           THETA,VARJ2,YIELD)                                     RSDU 100
      CURYS=UNIAX+EPSTN(KGAUS)*HARDS                                    RSDU 101
      BRING=1.0                                                         RSDU 102
      IF(YIELD.GT.CURYS) BRING=CURYS/YIELD                              RSDU 103
      DO 130 ISTR1=1,NSTR1                                              RSDU 104
  130 STRSG(ISTR1,KGAUS)=BRING*SGTOT(ISTR1)                             RSDU 105
      EFFST(KGAUS)=BRING*YIELD                                          RSDU 106
C*** ALTERNATIVE LOCATION OF STRESS REDUCTION LOOP TERMINATION CARD     RSDU 107
C  90 CONTINUE                                                          RSDU 108
C***                                                                    RSDU 109
      GO TO 190                                                         RSDU 110
   60 DO 180 ISTR1=1,NSTR1                                              RSDU 111
  180 STRSG(ISTR1,KGAUS)=STRSG(ISTR1,KGAUS)+DESIG(ISTR1)                RSDU 112
      EFFST(KGAUS)=YIELD                                                RSDU 113
C                                                                       RSDU 114
C*** CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE       RSDU 115
C    ELEMENT NODES                                                      RSDU 116
  190 MGASH=0                                                           RSDU 117
      DO 140 INODE=1,NNODE                                              RSDU 118
      DO 140 IDOFN=1,NDOFN                                              RSDU 119
      MGASH=MGASH+1                                                     RSDU 120
      DO 140 ISTRE=1,NSTRE                                              RSDU 121
  140 ELOAD(IELEM,MGASH)=ELOAD(IELEM,MGASH)+BMATX(ISTRE,MGASH)*         RSDU 122
     .STRSG(ISTRE,KGAUS)*DVOLU                                          RSDU 123
   40 CONTINUE                                                          RSDU 124
   20 CONTINUE                                                          RSDU 125
      RETURN                                                            RSDU 126
      END                                                               RSDU 127
```

RSDU 18–19	Compute $\sqrt{(3)}$ and 2π.
RSDU 20–22	Zero the array in which the equivalent nodal forces, calculated in Step h, will be stored.
RSDU 23	Zero the Gauss point counter over all elements.
RSDU 24	Loop over each element.
RSDU 25	Identify the element material property number.
RSDU 26–28	Identify the initial uniaxial yield stress, $\sigma_Y{}^\circ$ (or c for Mohr–Coulomb or Drucker–Prager criteria), the linear strain hardening parameter H' and the friction angle ϕ for Mohr–Coulomb and Drucker–Prager materials.
RSDU 29	For a Mohr–Coulomb material evaluate the equivalent yield stress as $c \cos\phi$.
RSDU 30–31	For a Drucker–Prager material evaluate the equivalent yield stress as k' according to (7.18).
RSDU 36–42	Store the element nodal coordinates in array ELCOD and the nodal displacements due to the application of the residual forces in array ELDIS.

RSDU 43	Evaluate the elastic $\boldsymbol{D}$ matrix.
RSDU 44	Identify the element thickness.
RSDU 45	Zero the local Gauss point counter.
RSDU 46–49	Enter the loops for numerical integration and evaluate the local coordinates (ξ, η) at the sampling point.
RSDU 50–51	Increment the local and global Gauss point counters.
RSDU 52	Evaluate the shape functions N_i and their derivatives $\partial N_i/\partial\xi$, $\partial N_i/\partial\eta$.
RSDU 53–54	Evaluate the Gauss point coordinates GPCOD(IDIME, KGASP), the determinant of the Jacobian matrix $\lvert\boldsymbol{J}\rvert$ and the Cartesian derivatives of the shape functions $\partial N_i/\partial_x$, $\partial N_i/\partial y$ (or $\partial N_i/\partial r$, $\partial N_i/\partial z$ for axisymmetric problems).
RSDU 55–57	Calculate the elemental volume for numerical integration as $\lvert\boldsymbol{J}\rvert W_\xi W_\eta$ taking care to multiply by the appropriate thickness or by $2\pi r$ for axisymmetric problems. The default value of the thickness is 1.0.
RSDU 58	Compute the strain matrix $\boldsymbol{B}$ for the Gauss point.
RSDU 59–60	Compute the stress increment STRES(ISTR1), assuming elastic behaviour as $d\boldsymbol{\sigma}_e{}^r = \boldsymbol{D}d\boldsymbol{\epsilon}^r$.
RSDU 61	Compute the yield stress for the $(r-1)^{\text{th}}$ iteration as $\sigma_Y{}^\circ + H'\bar{\epsilon}_p{}^{r-1}$.
RSDU 62–64	Store $d\boldsymbol{\sigma}_e{}^r$ as DESIG(ISTR1) and $\boldsymbol{\sigma}_e{}^r$ as SIGMA(ISTR1).
RSDU 65–66	Evaluate the effective stress in Col. 3, Table 7.2 and store as YIELD.
RSDU 67–68	Check if the Gauss point had yielded on the previous iteration, i.e. if $\bar{\sigma}^{r-1} > \sigma_Y{}^\circ + H'\bar{\epsilon}_p{}^{r-1}$ which is the first operation of Step *d*.
RSDU 69–70	If the Gauss point was previously elastic, check to see if it has yielded during this iteration.
RSDU 71	For a Gauss point which yields during the iteration calculate $$R = \frac{\bar{\sigma}_e{}^r - \sigma_Y}{\bar{\sigma}_e{}^r - \bar{\sigma}^{r-1}}.$$
RSDU 73–74	Check to see if a Gauss point which had previously yielded is unloading during this iteration. If yes, go to 60.
RSDU 75	Otherwise, set $R = 1$.
RSDU 76–77	Evaluate the number of steps into which the excess stress, $Rd\boldsymbol{\sigma}_e{}^r$ is to be divided according to (7.94).
RSDU 78	Compute $(1-R)$.
RSDU 79–81	Compute $\boldsymbol{\sigma}^{r-1} + (1-R)d\boldsymbol{\sigma}_e{}^r$ according to Step *e* and store in SGTOT(ISTR1) and evaluate $Rd\boldsymbol{\sigma}_e{}^r/m$ and store in STRES(ISTR1).
RSDU 82	Loop over each stress reduction step.
RSDU 83–87	Compute the vectors $\boldsymbol{a}$ and $\boldsymbol{d}_D$.

RSDU 88–92 Compute $d\lambda$ according to (7.45) and store as DLAMD.

RSDU 93–96 Compute $\boldsymbol{\sigma}^r = \boldsymbol{\sigma}^{r-1} + (1-R)d\boldsymbol{\sigma}_e{}^r + Rd\boldsymbol{\sigma}_e{}^r/m - d\lambda \boldsymbol{d}_D/m$. When the summation process from 1 to m required in DO LOOP to index 90 is completed this will result in $\boldsymbol{\sigma}^r = \boldsymbol{\sigma}^{r-1} + d\boldsymbol{\sigma}_e{}^r - d\lambda \boldsymbol{d}_D$ to give the stress point E in Fig. 7.11.

RSDU 97 Compute the effective plastic strain as follows. From (7.51) we have

$$d\kappa = d\lambda \boldsymbol{a}^T \boldsymbol{\sigma} = \boldsymbol{\sigma}^T d\boldsymbol{\epsilon}_p,$$

or rewriting the right hand side in terms of the effective stress $\bar{\sigma}$ and effective plastic strain $\bar{\epsilon}_p$ we have

$$d\lambda \boldsymbol{a}^T \boldsymbol{\sigma} = \bar{\sigma} d\bar{\epsilon}_p,$$

and therefore

$$\bar{\epsilon}_p{}^r = \bar{\epsilon}_p{}^{r-1} + \frac{d\lambda\, \boldsymbol{a}^T \boldsymbol{\sigma}}{\bar{\sigma}}. \tag{7.96}$$

RSDU 98 Return to loop over the next stress reduction step. This statement is so placed that the final stresses $\boldsymbol{\sigma}^r$ are scaled down to lie on the yield surface only after all the reduction steps have been completed. An additional refinement can be introduced where, with reference to Fig. 7.11, the stresses are scaled to the yield surface after each reduction step. Such a refinement is not normally required; however it can be introduced by moving statement RSDU 98 to the position indicated in RSDU 108.

RSDU 99–100 Compute the effective stress $\bar{\sigma}^r$.

RSDU 101 Evaluate $\sigma_Y{}^\circ + H'\bar{\epsilon}_p{}^r$.

RSDU 102–105 Factor the stresses $\boldsymbol{\sigma}^r$ to ensure that they lie on the yield surface, according to $\boldsymbol{\sigma}^r = \boldsymbol{\sigma}^r(\sigma_Y{}^\circ + H'\bar{\epsilon}_p{}^r)/\bar{\sigma}^r$ as indicated in Fig. 7.11.

RSDU 106 Store the effective stress $\bar{\sigma}^r$ in array EFFST.

RSDU 108 Location of end of loop if the refinement indicated in RSDU 98 is to be included.

RSDU 111–113 For elastic Gauss points compute $\boldsymbol{\sigma}^r$ as $\boldsymbol{\sigma}^{r-1} + d\boldsymbol{\sigma}_e{}^r$ and store $\bar{\sigma}^r$ in EFFST.

RSDU 117–123 Compute the equivalent nodal forces as

$$(\boldsymbol{f}^{(e)})^r = \int_\Omega \boldsymbol{B}^T \boldsymbol{\sigma}^r \, d\Omega.$$

RSDU 124–125 Termination of loop for numerical integration and over each element respectively.

7.8.8 Subroutine OUTPUT

This subroutine outputs the results at a frequency determined by the output parameters NOUTP(1) and NOUTP(2) whose role is described in Section 6.5.3. The principal stresses and direction are also calculated in this subroutine and these are given by the following expressions

$$\sigma_{\max} = \frac{\sigma_x+\sigma_y}{2}+\sqrt{\left(\frac{(\sigma_x-\sigma_y)^2}{4}+\tau_{xy}{}^2\right)},$$

$$\sigma_{\min} = \frac{\sigma_x+\sigma_y}{2}-\sqrt{\left(\frac{(\sigma_x-\sigma_y)^2}{4}+\tau_{xy}{}^2\right)}$$

$$\theta = \tan^{-1}\left(\frac{2\tau_{xy}}{\sigma_x-\sigma_y}\right). \tag{7.97}$$

with x and y being replaced by r and z for the axisymmetric case. The term θ defines the angle which the maximum principal stress makes with the y (or z) axis; a positive angle being measured anticlockwise.

This subroutine is largely self-explanatory and is listed below.

```
      SUBROUTINE OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,
     .                  NOUTP,NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,
     .                  NTYPE,NCHEK)
C***********************************************************************
C
C**** THIS SUBROUTINE OUTPUTS DISPLACEMENTS.REACTIONS AND STRESSES
C
C***********************************************************************
      DIMENSION  NOFIX(MVFIX),NOUTP(2),STRSG(4,MTOTG),STRSP(3),
     .           TDISP(MTOTV),TREAC(MVFIX,2),EPSTN(MTOTG)
      KOUTP=NOUTP(1)
      IF(IITER.GT.1) KOUTP=NOUTP(2)
      IF(IITER.EQ.1.AND.NCHEK.EQ.0) KOUTP=NOUTP(2)
C
C*** OUTPUT DISPLACEMENTS
C
      IF(KOUTP.LT.1) GO TO 10
      WRITE(6,900)
  900 FORMAT(1H0,5X,13HDISPLACEMENTS)
      IF(NTYPE.NE.3) WRITE(6,950)
  950 FORMAT(1H0,6X,4HNODE,6X,7HX-DISP.,7X,7HY-DISP.)
      IF(NTYPE.EQ.3) WRITE(6,955)
  955 FORMAT(1H0,6X,4HNODE,6X,7HR-DISP.,7X,7HZ-DISP.)
      DO 20 IPOIN=1,NPOIN
      NGASH=IPOIN*2
      NGISH=NGASH-2+1
   20 WRITE(6,910) IPOIN,(TDISP(IGASH),IGASH=NGISH,NGASH)
  910 FORMAT(I10,3E14.6)
   10 CONTINUE
C
C*** OUTPUT REACTIONS
C
      IF(KOUTP.LT.2) GO TO 30
      WRITE(6,920)
  920 FORMAT(1H0,5X,9HREACTIONS)
      IF(NTYPE.NE.3) WRITE(6,960)
```

```
  960 FORMAT(1H0,6X,4HNODE,6X,7HX-REAC.,7X,7HY-REAC.)                   OTPT  37
      IF(NTYPE.EQ.3) WRITE(6,965)                                       OTPT  38
  965 FORMAT(1H0,6X,4HNODE,6X,7HR-REAC.,7X,7HZ-REAC.)                   OTPT  39
      DO 40 IVFIX=1,NVFIX                                               OTPT  40
   40 WRITE(6,910) NOFIX(IVFIX),(TREAC(IVFIX,IDOFN),IDOFN=1,2)          OTPT  41
   30 CONTINUE                                                          OTPT  42
C                                                                       OTPT  43
C*** OUTPUT STRESSES                                                    OTPT  44
C                                                                       OTPT  45
      IF(KOUTP.LT.3) GO TO 50                                           OTPT  46
      IF(NTYPE.NE.3) WRITE(6,970)                                       OTPT  47
  970 FORMAT(1H0,1X,4HG.P.,6X,9HXX-STRESS,5X,9HYY-STRESS,5X,9HXY-STRESS,OTPT  48
     .5X,9HZZ-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,5HANGLE,3X,         OTPT  49
     . 6HE.P.S.)                                                        OTPT  50
      IF(NTYPE.EQ.3) WRITE(6,975)                                       OTPT  51
  975 FORMAT(1H0,1X,4HG.P.,6X,9HRR-STRESS,5X,9HZZ-STRESS,5X,9HRZ-STRESS,OTPT  52
     .5X,9HTT-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,5HANGLE,3X,         OTPT  53
     . 6HE.P.S.)                                                        OTPT  54
      KGAUS=0                                                           OTPT  55
      DO 60 IELEM=1,NELEM                                               OTPT  56
      KELGS=0                                                           OTPT  57
      WRITE(6,930) IELEM                                                OTPT  58
  930 FORMAT(1H0,5X,13HELEMENT NO. =,I5)                                OTPT  59
      DO 60 IGAUS=1,NGAUS                                               OTPT  60
      DO 60 JGAUS=1,NGAUS                                               OTPT  61
      KGAUS=KGAUS+1                                                     OTPT  62
      KELGS=KELGS+1                                                     OTPT  63
      XGASH=(STRSG(1,KGAUS)+STRSG(2,KGAUS))*0.5                         OTPT  64
      XGISH=(STRSG(1,KGAUS)-STRSG(2,KGAUS))*0.5                         OTPT  65
      XGESH=STRSG(3,KGAUS)                                              OTPT  66
      XGOSH=SQRT(XGISH*XGISH+XGESH*XGESH)                               OTPT  67
      STRSP(1)=XGASH+XGOSH                                              OTPT  68
      STRSP(2)=XGASH-XGOSH                                              OTPT  69
      IF(XGISH.EQ.0.0) XGISH=0.1E-20                                    OTPT  70
      STRSP(3)=ATAN(XGESH/XGISH)*28.647889757                           OTPT  71
   60 WRITE(6,940) KELGS,(STRSG(ISTR1,KGAUS),ISTR1=1,4),                OTPT  72
     . (STRSP(ISTRE),ISTRE=1,3),EPSTN(KGAUS)                            OTPT  73
  940 FORMAT(I5,2X,6E14.6,F8.3,E14.6)                                   OTPT  74
   50 CONTINUE                                                          OTPT  75
      RETURN                                                            OTPT  76
      END                                                               OTPT  77
```

OTPT 11–13 Set the output indicator, KOUTP, according to whether or not this is the first iteration of a load increment or not. If it is the first iteration the results will be output according to NOUTP(1) but for a converged solution the results are output according to NOUTP(2).

OTPT 17–29 For an output code value of 1 or greater, output the nodal displacements after printing the appropriate headings.

OTPT 33–42 For an output code of 2 or greater, output appropriate headings and the reactions at restrained nodal points.

OTPT 46 For an output code of 3 output the Gauss point stresses.

OTPT 47–54 Write appropriate headings.

OTPT 56–59 Loop over each element and write the element number.

OTPT 60–61 Loop over each element Gauss point.

OTPT 62–71 Evaluate the principal stresses and direction for each Gauss point according to (7.97).

OTPT 72–74 Output the Cartesian stress components, the principal stresses and direction and the total effective plastic strain for each Gauss point. This latter quantity gives an immediate indication whether the Gauss point has yielded or not, since it will be zero for all elastic points.

7.8.9 The main, master or controlling segment

This segment controls the calling, in order, of the other subroutines and is similar in structure to the segment described in Section 3.8 for one-dimensional situations. Its other function is to control the iterative process and also the incrementing of the applied loads.

The following channel numbers are employed by the program: 5 (card reader), 6 (line printer), 1, 2, 3, 4, 8 (scratch files).

This routine is self-explanatory and is presented below without further comment.

```
      MASTER PLAST
C**********************************************************************
C     PROGRAM FOR THE ELASTO-PLASTIC ANALYSIS OF PLANE STRESS,
C     PLANE STRAIN AND AXISYMMETRIC SOLIDS
C**********************************************************************
      DIMENSION ASDIS(300),COORD(150,2),ELOAD(40,18),ESTIF(18,18),
     .          EQRHS(10),EQUAT(80,10),FIXED(300),GLOAD(80),GSTIF(3240),
     .          IFFIX(300),LNODS(40,9),LOCEL(18),MATNO(40),
     .          NACVA(80),NAMEV(10),NDEST(18),NDFRO(40),NOFIX(30),
     .          NOUTP(2),NPIVO(10),
     .          POSGP(4),PRESC(30,2),PROPS(5,7),RLOAD(40,18),
     .          STFOR(300),TREAC(30,2),VECRV(80),WEIGP(4),
     .          STRSG(4,360),TDISP(300),TLOAD(40,18),
     .          TOFOR(300),EPSTN(360),EFFST(360)
C
C*** PRESET VARIABLES ASSOCIATED WITH DYNAMIC DIMENSIONING
C
      CALL DIMEN(MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,MSTIF,MTOTG,MTOTV,
     .           MVFIX,NDOFN,NPROP,NSTRE)
C
C*** CALL THE SUBROUTINE WHICH READS MOST OF THE PROBLEM DATA
C
      CALL INPUT(COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,MMATS,
     .           MPOIN,MTOTV,MVFIX,NALGO,
     .           NCRIT,NDFRO,NDOFN,NELEM,NEVAB,NGAUS,NGAU2,
     .           NINCS,NMATS,NNODE,NOFIX,NPOIN,NPROP,NSTRE,
     .           NSTR1,NTOTG,NTOTV,
     .           NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP)
C
C*** CALL THE SUBROUTINE WHICH COMPUTES THE CONSISTENT LOAD VECTORS
C    FOR EACH ELEMENT AFTER READING THE RELEVANT INPUT DATA
C
      CALL LOADPS(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
     .            NEVAB,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
     .            PROPS,RLOAD,WEIGP,NDOFN)
C
C*** INITIALISE CERTAIN ARRAYS
C
```

```
      CALL ZERO(ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM,
     .          NEVAB,NGAUS,NSTR1,NTOTG,EPSTN,EFFST,
     .          NTOTV,NVFIX,STRSG,TDISP,TFACT,
     .          TLOAD,TREAC,MVFIX)
C
C*** LOOP OVER EACH INCREMENT
C
      DO 100 IINCS = 1,NINCS
C
C*** READ DATA FOR CURRENT INCREMENT
C
      CALL INCREM(ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,MTOTV,
     .            MVFIX,NDOFN,NELEM,NEVAB,NOUTP,NOFIX,NTOTV,
     .            NVFIX,PRESC,RLOAD,TFACT,TLOAD,TOLER)
C
C*** LOOP OVER EACH ITERATION
C
      DO 50 IITER = 1,MITER
C
C*** CALL ROUTINE WHICH SELECTS SOLUTION ALORITHM VARIABLE KRESL
C
      CALL ALGOR(FIXED,IINCS,IITER,KRESL,MTOTV,NALGO,
     .           NTOTV)
C
C*** CHECK WHETHER A NEW EVALUATION OF THE STIFFNESS MATRIX IS REQUIRED
C
      IF(KRESL.EQ.1) CALL STIFFP(COORD,EPSTN,IINCS,LNODS,MATNO,
     .          MEVAB,MMATS,MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE,
     .         NSTRE,NSTR1,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     .          STRSG,NTYPE,NCRIT)

C
C*** SOLVE EQUATIONS
C
      CALL FRONT(ASDIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,IITER,
     .           GLOAD,GSTIF,LOCEL,LNODS,KRESL,MBUFA,MELEM,MEVAB,MFRON,
     .           MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,NDOFN,NELEM,NEVAB,
     .           NNODE,NOFIX,NPIVO,NPOIN,NTOTV,TDISP,TLOAD,TREAC,
     .           VECRV)
C
C*** CALCULATE RESIDUAL FORCES
C
      CALL RESIDU(ASDIS,COORD,EFFST,ELOAD,FACTO,IITER,LNODS,
     .            LPROP,MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,NDOFN,
     .            NELEM,NEVAB,NGAUS,NNODE,NSTR1,NTYPE,POSGP,PROPS,
     .            NSTRE,NCRIT,STRSG,WEIGP,TDISP,EPSTN)
C
C*** CHECK FOR CONVERGENCE
C
      CALL CONVER(ELOAD,IITER,LNODS,MELEM,MEVAB,MTOTV,NCHEK,NDOFN,
     .            NELEM,NEVAB,NNODE,NTOTV,PVALU,STFOR,TLOAD,TOFOR,TOLER)
C
C*** OUTPUT RESULTS IF REQUIRED
C
      IF(IITER.EQ.1.AND.NOUTP(1).GT.0)
     .CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .            NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK)
C
C*** IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS
C
      IF(NCHEK.EQ.0) GO TO 75
   50 CONTINUE
C
C***
C
```

```
      IF(NALGO.EQ.2) GO TO 75
      STOP
   75 CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .            NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK)
  100 CONTINUE
      STOP
      END
```

7.9 Numerical examples

The first numerical example considered is illustrated in Fig. 7.12(a). The problem studied is that of a thick cylinder subjected to a gradually increasing internal pressure, with plane strain conditions being assumed in the axial direction. A Von Mises yield criterion is assumed and the numerical solutions obtained compared with the theoretical results of Reference 14. The pressure/radial displacement characteristics are shown in Fig. 7.12(b) and good

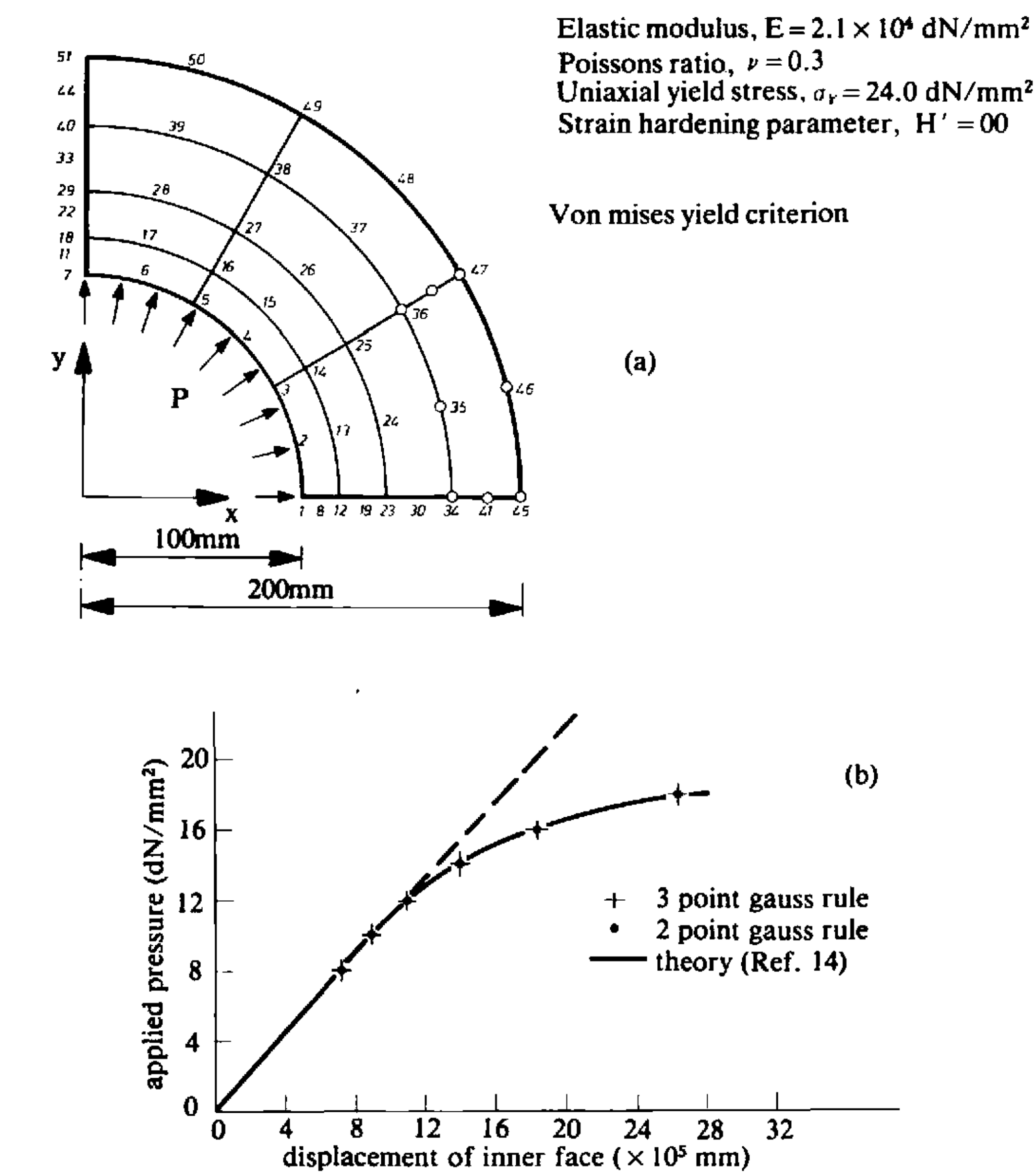


Fig. 7.12 (a) Mesh and material properties employed in the elasto-plastic analysis of an internally pressurised thick cylinder under plane strain conditions. (b) Displacement of the inner surface with increasing pressure for the problem of Fig. 7.12(a).

agreement between the numerical and analytical solutions is evident. In the numerical studies, collapse was deemed to have occurred if the iterative procedure diverged for an incremental load increase.

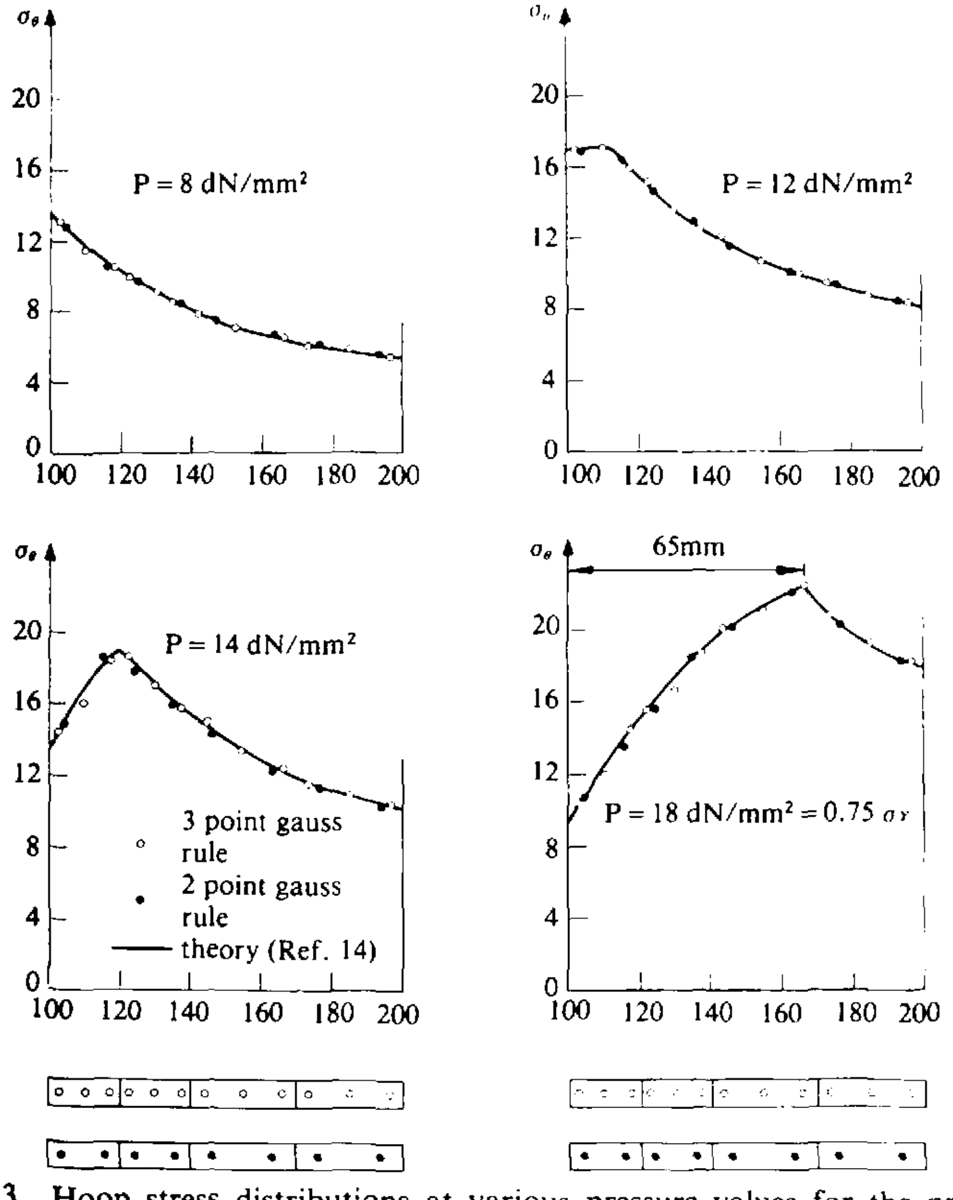


Fig. 7.13 Hoop stress distributions at various pressure values for the problem of Fig. 7.12(a).

Fig. 7.13 shows the circumferential (hoop) stress distributions for specified pressure values. Again a good agreement is evident. In solution both a two-point and three-point Gaussian integration rule was considered. Whilst the nodal displacements obtained by use of both rules are practically identical, it is seen from Fig. 7.13 that use of a 2×2 integrating rule gives superior stress values to a 3×3 rule. This is a general result for elasto-plastic problems and therefore use of a two-point rule is recommended. This phenomenon is an example of the benefit of a reduced integration order for parabolic isoparametric elements.[15]

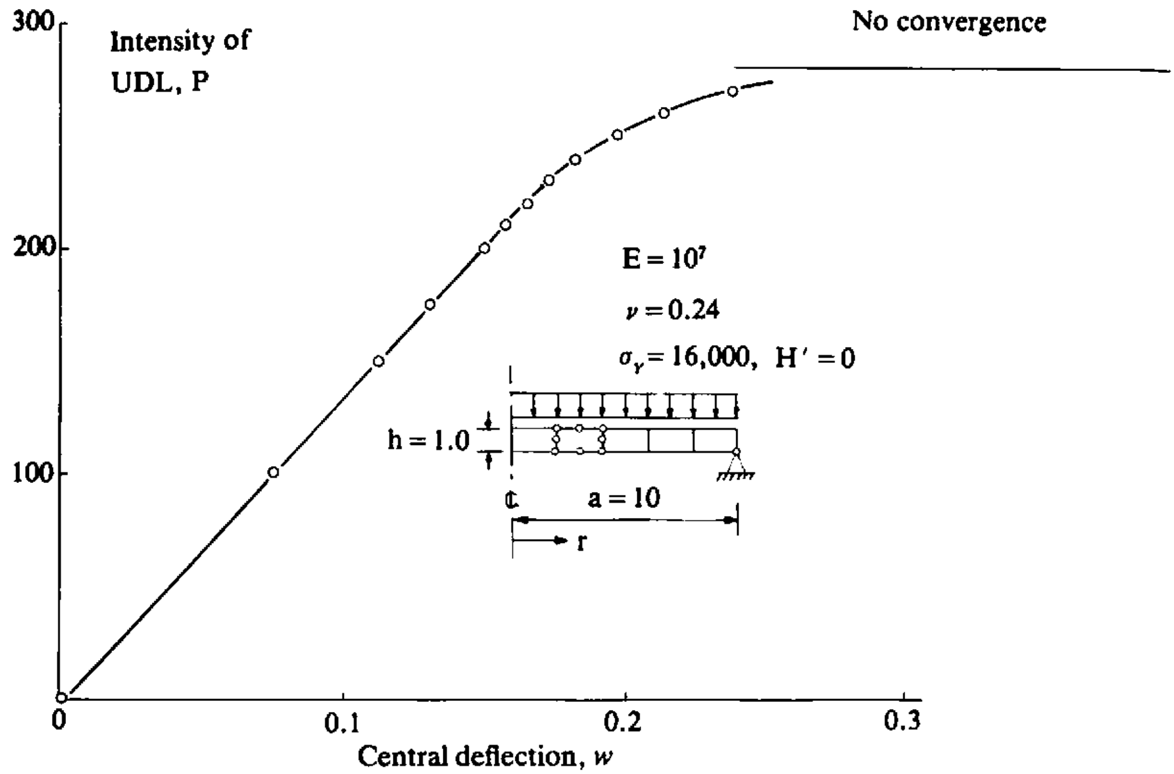


Fig. 7.14 Load/central deflection response for a uniformly loaded simply supported circular plate.

The second example considered is the simply supported circular plate shown in Fig. 7.14.

The plate is modelled by five axisymmetric elements and the loading takes the form of a progressively increasing uniformly distributed load. The growth in central deflection with increasing load is shown in Fig. 7.14. A converged solution was obtained for $P = 270$ but the numerical process diverged for $P = 280$ and consequently the collapse load is taken to be 270. This is in good agreement with the value of 260 quoted in Ref. 16, particularly in view of the coarse mesh employed in the present study. Fig. 7.15 shows the deflection profile with increasing applied load.

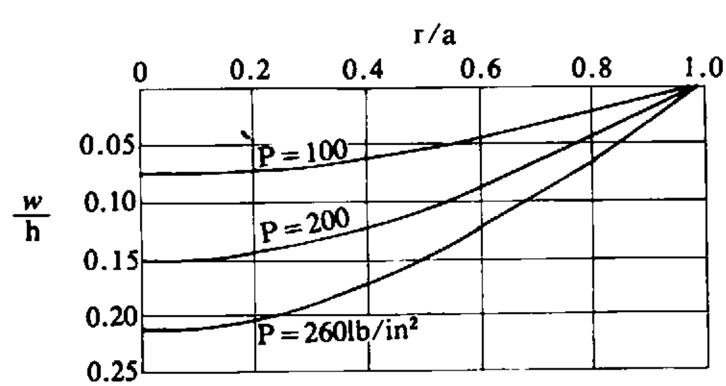


Fig. 7.15 Deflection profiles for the problem of Fig. 7.14 at various applied load values.

7.10 Problems

7.1 In Section 7.2.1 it was stated that the Von Mises law implies that yielding begins when the (recoverable) elastic energy of distortion, D, reaches a critical value. Prove this by showing that J_2' is proportional to D, since D can be written as

$$D = \tfrac{1}{2}\sigma_{ij}\,\epsilon_{ij} - \frac{(1-2\nu)}{12\mu(1+\nu)}(\sigma_{ii})^2. \tag{7.98}$$

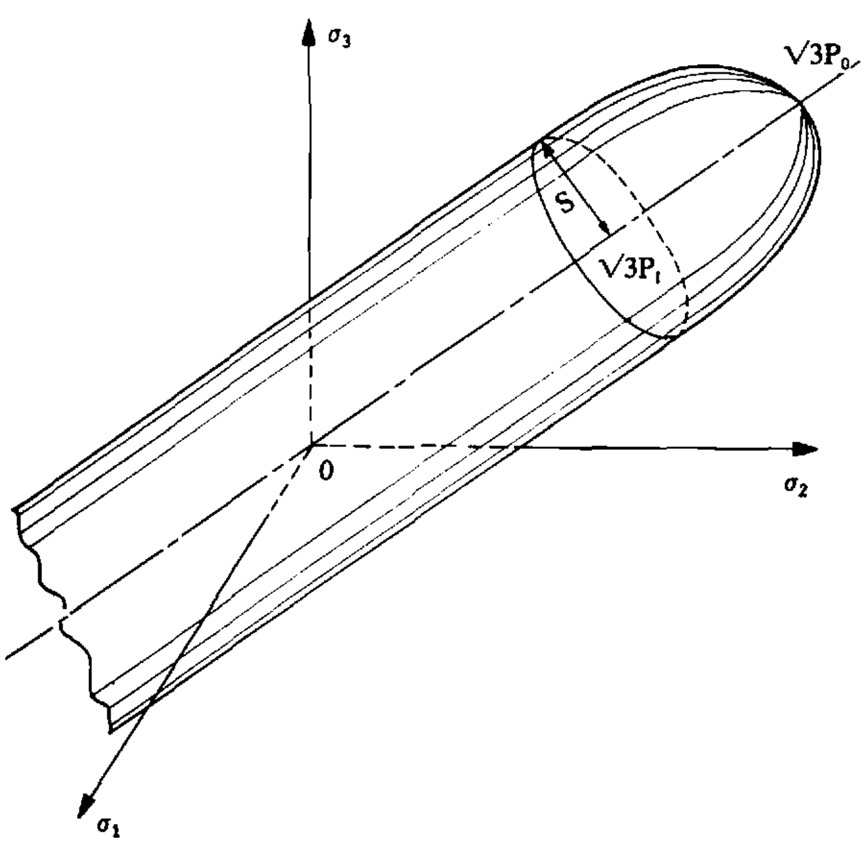


Fig. 7.16 Geometric representation of the Berg yield criterion—Problem 7.2.

7.2 A yield criterion has been proposed by Berg[17] which attempts to account for the tensile failure of a material due to the formation of voids at a sufficiently high strain level. The yield surface is illustrated in Fig. 7.16 and can be seen to be made up of two distinct portions. For stress levels below a mean hydrostatic tension of P_I the material yields according to the Von Mises cylinder of radius S. The yield surface in the tensile range is terminated by an elliptic cap whose extremity is defined by P_0. The three constants S, P_I and P_0 are material constants and must be experimentally determined. The two distinct portions of the yield surface can be expressed as

$$\sqrt{2}(J_2')^{\frac{1}{2}} = S \qquad \text{for } \sigma_m \leqslant P_I$$

$$[2J_2' + H(\sigma_m - P_I)^2]^{\frac{1}{2}} = S \qquad P_I \leqslant \sigma_m \leqslant P_0, \tag{7.99}$$

where $H = S^2/(P_I - P_0)^2$ and σ_m is the mean hydrostatic pressure.

Show that this yield criterion can be expressed in the form of three constants C_1, C_2 and C_3 as indicated in Section 7.4 where

$$C_1 = 0, \quad C_2 = \sqrt{2}, \quad C_3 = 0 \qquad \text{for} \qquad \sigma_m \leqslant P_I$$

$$C_1 = H(\sigma_m - P_I)/S, \quad C_2 = 2J_2'/S, \quad C_3 = 0 \qquad P_I \leqslant \sigma_m \leqslant P_0.$$

7.3 A certain material yields when the maximum principal stress reaches a critical value, Y. Assuming identical behaviour in tension and compression, determine the geometrical form of the yield surface. The solution is given in Fig. 7.17.

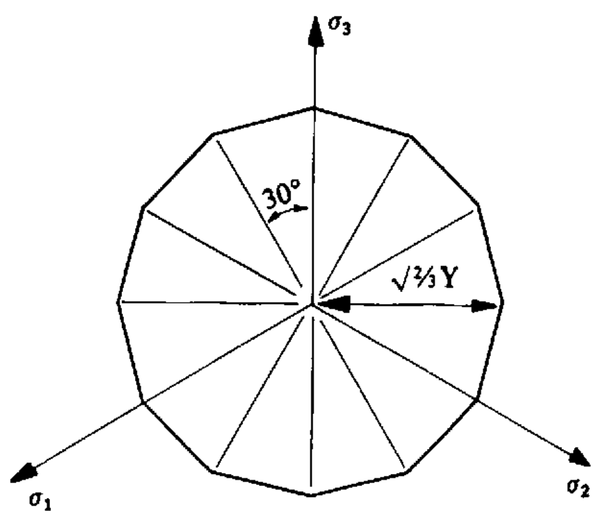


Fig. 7.17 π plane representation of a yield criterion based on maximum principal stress values—Problem 7.3.

7.4 The assumption of a linear strain hardening material law may prove to be inadequate for certain situations. If the uniaxial stress/strain test curve for the material is known, then it is possible to represent the stress-plastic strain relationship in a piecewise linear fashion as shown in Fig. 7.18 and the instantaneous yield stress can be written in the form $\sigma_Y = \sigma_Y{}^0 + S(\bar{\epsilon}_p)$ where $S(\bar{\epsilon}_p)$ is the piecewise linear function describing the increase (or decrease) in the initial yield stress $\sigma_Y{}^0$ with the increase of effective plastic strain $\bar{\epsilon}_p$. The program modifications required to describe this behaviour will all be included in subroutine RESIDU, except for changes in material property specification which will need to be made in subroutine INPUT. Carry out all necessary modifications.

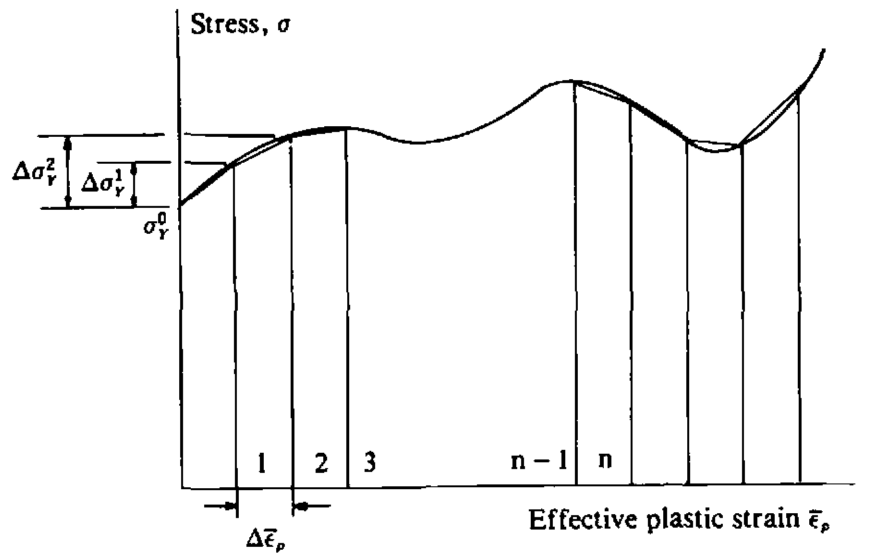


Fig. 7.18 Piecewise-linear representation of material strain hardening—Problem 7.4.

7.5 By using the mesh of Fig. 7.12(a) and solving as an axisymmetric problem, use program PLANET (documented in Appendix II, Section A2.1) to determine the elasto-plastic stress and displacement distributions in a sphere when it is loaded by an incrementally applied internal pressure. The dimensions and material properties of the sphere are given by reference to Fig. 7.12. Assume a Tresca yield criterion for solution and compare your results with the solution given in Ref. 1.

7.6 Use program PLANET to solve the problem illustrated in Fig. 1.2, Chapter 1. Use both a Tresca and Von Mises yield criterion and compare the plastic zone distributions obtained with those of Fig. 1.2.

7.7 Subroutine CONVER, described in Section 6.5.4, bases convergence of the nonlinear solution process on the *global* norm of the residual force vector. Modify subroutine CONVER so that convergence is based on expression (3.27) in which the summation signs are absent; so that convergence is monitored *locally* at each of the nodes 1 to N in turn.

7.8 Modify subroutine CONVER, Section 6.5.4 so that convergence is monitored locally at each node according to the displacement changes that occur during a particular iteration, r, as follows.

$$\frac{|\Delta \boldsymbol{d}^r|}{|\boldsymbol{d}^1|}\times 100 \leqslant \text{TOLER}, \tag{7.100}$$

where $\boldsymbol{d}^1$ is the elastic displacement occurring upon application of the load increment and $\Delta\boldsymbol{d}^r$ is the change in nodal displacement during the r^{th} iteration.

7.9 Modify program PLANET to undertake the elasto-plastic solution of three-dimensional solids. To simplify the task consider only the Von Mises yield criterion and assume that the solid is loaded by nodal point loads only.

7.10 The yield criterion to be employed in program PLANET is specified by means of control parameter NCRIT in subroutine INPUT described in Section 6.5.1. In some applications, such as steel-concrete composites, it is necessary to employ a different yield surface for different parts of the structure. Modify program PLANET so that the yield criterion governing elasto-plastic behaviour is separately specified for each element in the solid.

7.11 References

1. HILL, R., *The Mathematical Theory of Plasticity*, Oxford University Press, 1950.
2. PRAGER, W., *An Introduction to Plasticity*, Addison-Wesley, Amsterdam and London, 1959.
3. HOFFMAN, O. and SACHS, G., *Introduction to the Theory of Plasticity for Engineers*, McGraw-Hill, 1953.
4. BRIDGMAN, P. W., *Studies in Large Plastic Flow and Fracture*, McGraw-Hill, New York, 1952.
5. BISHOP, A. W., The strength of soils as engineering materials, *Geotechnique*, **16,** 89–130 (1966).
6. DAVIS, E. M., Theories of plasticity and the failure of soil masses, Ch. 6 *Soil Mechanics*, Ed. I. K. Lee, Butterworths, London, 1968.
7. ZIENKIEWICZ, O. C., VALLIAPPAN, S. and KING, I. P., Elasto-plastic solutions of engineering problems; Initial stress finite element approach, *Int. J. Num. Meth. Engng.* **1,** 75–100 (1969).
8. YAMADA, Y., YOSHIMURA, N. and SAKURAI, T., Plastic stress–strain matrix and its application for the solution of elastic-plastic problems by Finite Element Method, *Int. J. Mech. Sci.* **10,** 343–354 (1968).
9. BLAND, D. R., The associated flow rule of plasticity, *J. Mech. Phy. of Solids*, **6,** 71–78 (1957).
10. NAYAK, G. C. and ZIENKIEWICZ, O. C., Convenient form of stress invariants for Plasticity, *Journ. of the Struct. Div. Proc. of A.S.C.E.*, 949–953, April 1972.
11. FREDERICK, D. and CHANG, T. S., *Continuum Mechanics*, Allyn and Bacon, 1965.
12. KOITER, W. T., Stress–strain relations, uniqueness and variational theorems for elastic-plastic materials with singular yield surface, *Quart. Appl. Math.*, **11,** 350–354 (1953).
13. HINTON, E. and OWEN, D. R. J., *Finite Element Programming*, Academic Press, London, 1977.
14. HODGE, P. G. and WHITE, G. N., A quantitative comparison of flow and deformation theories of plasticity' *J. Appl. Mech.* **17,** 180–184 (1950).
15. ZIENKIEWICZ, O. C. and HINTON, E., Reduced integration, function smoothing and non-conformity in finite element analysis (with special reference to thick plates), *J. of the Franklin Institute*, **302,** Nos. 5 and 6, Nov./Dec. 1976.
16. ARMEN, H., Jr., PIFKO, A. and LEVINE, H. S., Finite element analysis of structures in the plastic range, *N.A.S.A. Contractor Report*, CR.1649 (1971).

17. Berg, C. A., Plastic dilation and void interaction, Battelle Inst. Material Science Colloquia, Sept. 1969. *Inelastic Behaviour of Solids*, Ed. M. F. Kanninen *et al.*, McGraw-Hill, 171–210, 1970.
18. Krieg, R. D. and Krieg, D. B., Accuracies of numerical solution methods for the elastic-perfectly plastic mode, *Trans. ASME, J. Pressure Vessel Technology*, **99**, 4, 510–515 (1977).
19. Schreyer, H. L., Kulak, R. F. and Kramer, J. M., Accurate numerical solutions for elastic-plastic models, *Trans. ASME, J. Pressure Vessel Technology*, **101**, 3, 226–234 (1979).
20. Samuelsson, A. and Fröier, M., Finite elements in plasticity—A variational inequality approach, in *The Mathematics of Finite Elements and Applications III*, Ed. J. R. Whiteman, Academic Press, 1979.

Chapter 8
Elasto-viscoplastic problems in two dimensions

8.1 Introduction

In all inelastic deformations time rate effects are always present to some degree. Whether or not their exclusion has a significant influence on the prediction of the material behaviour depends upon several factors. In the study of structural components under static loading conditions at normal temperatures it is accepted that time rate effects are generally not important and the conventional theory of plasticity, as described in Chapter 7, then models the behaviour adequately. However metals, especially under high temperatures, exhibit simultaneously the phenomena of creep and visco-plasticity. The former is essentially a redistribution of stress and/or strains with time under elastic material response while the latter is a time dependent plastic deformation. Experimental observations cannot distinguish between the two phenomena and their separation has been largely an analytical convenience rather than a physical requirement. Numerical processes, as described in this chapter, allow the simultaneous description of both effects.

A further situation in which time rate effects are important is in the dynamic transient loading of structures. For example, it can be experimentally demonstrated that the instantaneous yield stress of materials under high strain rates can be significantly greater than the corresponding quasi-static value. This class of problem is dealt with in Chapter 10.

In this chapter we utilise the theory of viscoplasticity to provide a unified approach to problems of creep and plasticity. As well as providing solutions to time-dependent situations the viscoplastic algorithm can provide economic solution for classic elasto-plastic problems since it can be readily shown that the steady-state solution of the viscoplastic problem is identical to the corresponding conventional static elasto-plastic solution. Furthermore, by reducing the yield stress of the material to zero, elastic creep problems can be solved.

The concept of 'overlay models' is also introduced in this chapter. In this, the solid is assumed, for mathematical convenience only, to be composed of several layers or overlays each of which undergo the same deformation. By assigning different properties to each overlay a composite behaviour can be obtained which exhibits all the essential characteristics of the visco-elastic-plastic response of many real materials.

The basic one-dimensional rheological model developed in Chapter 4 is now extended to the case of a general continuum and the essential steps employed in the numerical solution algorithm are discussed. Since most of the matrix expressions involved in viscoplastic analysis are common to conventional elasto-plastic theory, the majority of the subroutines developed in Chapter 7 can be again used with little or no change. The additional subroutines required are then constructed and assembled to form a working program. Finally it is briefly demonstrated how the overlay principle can be used to simulate a complex material response.

8.2 Theory of elasto-viscoplastic solids

8.2.1 Basic expressions

In the usual manner for nonlinear continua problems it is assumed that the total strain, $\boldsymbol{\epsilon}$, can be separated into elastic, $\boldsymbol{\epsilon}_e$, and viscoplastic, $\boldsymbol{\epsilon}_{vp}$, components, so that the total strain rate can be expressed as[1–3]

$$\dot{\boldsymbol{\epsilon}} = \dot{\boldsymbol{\epsilon}}_e + \dot{\boldsymbol{\epsilon}}_{vp}, \tag{8.1}$$

where $(\dot{\ })$ represents differentiation with respect to time. The total stress rate depends on the elastic strain rate according to

$$\dot{\boldsymbol{\sigma}} = \boldsymbol{D}\dot{\boldsymbol{\epsilon}}_e, \tag{8.2}$$

where $\boldsymbol{D}$ is the elasticity matrix. The onset of viscoplastic behaviour is governed by a scalar yield condition of the form

$$F(\boldsymbol{\sigma}, \boldsymbol{\epsilon}_{vp}) - F_0 = 0, \tag{8.3}$$

in which F_0 is the uniaxial yield stress which may itself be a function of a hardening parameter, κ. For frictional materials F_0 is the equivalent yield stress as given by Column 4, Table 7.2. It is assumed that viscoplastic flow occurs for values of $F > F_0$ only.

It is now necessary to choose a specific law defining the viscoplastic strains. The simplest option is one in which the viscoplastic strain rate depends only on the current stresses, so that

$$\dot{\boldsymbol{\epsilon}}_{vp} = f(\boldsymbol{\sigma}). \tag{8.4}$$

This relationship can be generalised to include strain hardening and temperature dependence and the influence of state dependent variables, such as damage parameters for rupture theories, can also be considered.

One explicit form of (8.4) which has wide applicability, is offered by the following viscoplastic flow rule.[4]

$$\dot{\boldsymbol{\epsilon}}_{vp} = \gamma\langle\Phi(F)\rangle\frac{\partial Q}{\partial \boldsymbol{\sigma}}, \tag{8.5}$$

in which $Q = Q(\boldsymbol{\sigma}, \boldsymbol{\epsilon}_{vp}, \kappa)$ is a 'plastic' potential and γ is a fluidity parameter controlling the plastic flow rate. The term $\Phi(x)$ is a positive monotonic increasing function for $x>0$ and the notation $\langle\ \rangle$ implies

$$\begin{aligned} \langle\Phi(x)\rangle &= \Phi(x) \text{ for } x>0 \\ \langle\Phi(x)\rangle &= 0 \qquad\quad x\leqslant 0. \end{aligned} \tag{8.6}$$

Comparison of (8.5) with (7.28) shows an analogy between the flow rule of conventional non-associated plasticity and the present definition of viscoplastic flow rate. If, once again, we restrict ourselves to associated plasticity situations, in which case $F \equiv Q$, expression (8.5) reduces to

$$\dot{\boldsymbol{\epsilon}}_{vp} = \gamma\langle\Phi(F)\rangle\frac{\partial F}{\partial \boldsymbol{\sigma}} = \gamma\langle\Phi\rangle\boldsymbol{a}, \tag{8.7}$$

where the same definition of the flow vector $\boldsymbol{a}$ is employed as in (7.42). Different choices have been recommended[5] for the function Φ. The two most common versions are

$$\Phi(F) = e^{M\left(\frac{F-F_0}{F_0}\right)} - 1, \tag{8.8}$$

and

$$\Phi(F) = \left(\frac{F-F_0}{F_0}\right)^N, \tag{8.9}$$

in which M and N are arbitrary prescribed constants. The latter option, when employed in (8.7) can be made to model the Norton power law of metallic creep by assigning the threshold uniaxial yield value, F_0, to zero (or to an arbitrarily small value for numerical convenience).

8.2.2 The viscoplastic strain increment

With the strain rate law expressed by (8.7) we can define a strain increment $\Delta\boldsymbol{\epsilon}_{vp}{}^n$ occurring in a time interval $\Delta t_n = t_{n+1} - t_n$ using an implicit time stepping scheme, as[6]

$$\Delta\boldsymbol{\epsilon}_{vp}{}^n = \Delta t_n[(1-\Theta)\dot{\boldsymbol{\epsilon}}_{vp}{}^n + \Theta\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1}]. \tag{8.10}$$

For $\Theta = 0$ we obtain the Euler time integration scheme which is also referred to as 'fully explicit' (or forward difference method) since the strain increment is completely determined from conditions existing at time, t_n. On the other hand $\Theta = 1$ gives a 'fully implicit' (or backward difference) scheme with the strain increment being determined from the strain rate corresponding to the end of the time interval. The case $\Theta = \frac{1}{2}$ results in the so-called 'implicit trapezoidal' scheme which is also known generally as the Crank-Nicolson rule in the context of linear equations.

To define $\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1}$ in (8.10) we can use a limited Taylor series expansion and write

$$\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1} = \dot{\boldsymbol{\epsilon}}_{vp}{}^{n} + \boldsymbol{H}^n \Delta\boldsymbol{\sigma}^n, \tag{8.11}$$

where

$$\boldsymbol{H}^n = \left(\frac{\partial \dot{\boldsymbol{\epsilon}}_{vp}}{\partial \boldsymbol{\sigma}}\right)^n = \boldsymbol{H}^n(\boldsymbol{\sigma}^n), \tag{8.12}$$

and $\Delta\boldsymbol{\sigma}^n$ is the stress change occurring in the time interval $\Delta t_n = t_{n+1} - t_n$. Thus (8.10) can be rewritten as

$$\Delta\boldsymbol{\epsilon}_{vp}{}^{n} = \dot{\boldsymbol{\epsilon}}_{vp}{}^{n}\Delta t_n + \boldsymbol{C}^n \Delta\boldsymbol{\sigma}^n, \tag{8.13}$$

where

$$\boldsymbol{C}^n = \Theta \Delta t_n \boldsymbol{H}^n. \tag{8.14}$$

We draw the attention of the reader to the fact that the matrix $\boldsymbol{H}$ defined in (8.12) is the matrix whose eigenvalues determine the limiting time step length, Δt_n which can be employed in the explicit integration schemes. The matrix $\boldsymbol{H}$ depends on the stress level and no difficulty arises in its evaluation and specific forms will be developed in Section 8.5.

8.2.3 Stress increments

Using the incremental form of (8.2) we obtain

$$\Delta\boldsymbol{\sigma}^n = \boldsymbol{D}\Delta\boldsymbol{\epsilon}_e{}^n = \boldsymbol{D}(\Delta\boldsymbol{\epsilon}^n - \Delta\boldsymbol{\epsilon}_{vp}{}^n). \tag{8.15}$$

Or expressing the total strain increment in terms of the displacement increment as

$$\Delta\boldsymbol{\epsilon}^n = \boldsymbol{B}^n \Delta\boldsymbol{d}^n, \tag{8.16}$$

and substituting for $\Delta\boldsymbol{\epsilon}_{vp}{}^n$ from (8.13), then (8.15) becomes

$$\Delta\boldsymbol{\sigma}^n = \hat{\boldsymbol{D}}^n(\boldsymbol{B}^n \Delta\boldsymbol{d}^n - \dot{\boldsymbol{\epsilon}}_{vp}{}^n \Delta t_n), \tag{8.17}$$

where

$$\hat{\boldsymbol{D}}^n = (\boldsymbol{I} + \boldsymbol{D}\boldsymbol{C}^n)^{-1}\boldsymbol{D} = (\boldsymbol{D}^{-1} + \boldsymbol{C}^n)^{-1}. \tag{8.18}$$

In (8.16) and (8.17) the notation $\boldsymbol{B}^n$ is employed to denote the possibility that the strain matrix may not be constant throughout the solution. For example, if large deformations are to be considered, the strain matrix for a Lagrangian formulation is nonlinear and can be written

$$\boldsymbol{B}^n = \boldsymbol{B}_0 + \boldsymbol{B}_{NL}{}^n, \tag{8.19}$$

where $\boldsymbol{B}_0$ represents the standard linear terms which do not vary during solution and $\boldsymbol{B}_{NL}{}^n$ contains the nonlinear quadratic terms. These latter expressions are dependent on the current displacements and therefore vary throughout the solution process.

The matrix $\boldsymbol{D}^n$ is a symmetric matrix when the visco-plastic law is associative. For the non-associated case, the matrix $\boldsymbol{C}^n$ is unsymmetric, requiring unsymmetric equation solvers for analysis.

For the solution of linear elastic problems by the explicit scheme ($\Theta = 0$), equation (8.17) simplifies considerably to give

$$\Delta\boldsymbol{\sigma}^n = \boldsymbol{D}(\boldsymbol{B}\Delta\boldsymbol{d}^n - \dot{\boldsymbol{\epsilon}}_{vp}{}^n \Delta t_n). \tag{8.20}$$

8.2.4 Equations of equilibrium

The equations of equilibrium to be satisfied at any instant of time, t_n, are

$$\int_\Omega [\boldsymbol{B}^n]^T \boldsymbol{\sigma}^n \, d\Omega + \boldsymbol{f}^n = \boldsymbol{0}, \tag{8.21}$$

where $\boldsymbol{f}^n$ is the vector of equivalent nodal loads due to applied surface tractions, body forces, thermal loads, etc. During a time increment the equilibrium equations which must be satisfied are given by the incremental form of (8.21) to be

$$\int_\Omega [\boldsymbol{B}^n]^T \Delta\boldsymbol{\sigma}^n \, d\Omega + \Delta\boldsymbol{f}^n = \boldsymbol{0}, \tag{8.22}$$

in which $\Delta\boldsymbol{f}^n$ represents the change in loads during the time interval Δt_n. In the majority of problems encountered in engineering the load increments are applied as discrete steps and thus $\Delta\boldsymbol{f}^n = \boldsymbol{0}$ for all time steps other than the first within an increment.

Using (8.13) and (8.20) the displacement increment occurring during time step Δt_n can be calculated as

$$\Delta\boldsymbol{d}^n = [\boldsymbol{K}_T{}^n]^{-1} \Delta\boldsymbol{V}^n$$

$$\Delta\boldsymbol{V}^n = \int_\Omega [\boldsymbol{B}^n]^T \hat{\boldsymbol{D}}^n \dot{\boldsymbol{\epsilon}}_{vp}{}^n \Delta t_n \, d\Omega + \Delta\boldsymbol{f}^n, \tag{8.23}$$

where $\boldsymbol{K}_T{}^n$ is the tangential stiffness matrix with the following form

$$\boldsymbol{K}_T{}^n = \int_\Omega [\boldsymbol{B}^n]^T \hat{\boldsymbol{D}}^n \boldsymbol{B}^n \, d\Omega, \tag{8.24}$$

and $\Delta\boldsymbol{V}^n$ are termed the *incremental pseudo-loads*. The displacement increments, $\Delta\boldsymbol{d}^n$, when substituted back into (8.20) give the stress increments

$\Delta\sigma^n$ and thus

$$\begin{aligned} \sigma^{n+1} &= \sigma^n + \Delta\sigma^n \\ d^{n+1} &= d^n + \Delta d^n. \end{aligned} \tag{8.25}$$

Use of (8.15) and (8.16) gives

$$\Delta\epsilon_{vp}{}^n = B^n \Delta d^n - D^{-1}\Delta\sigma^n, \tag{8.26}$$

and then

$$\epsilon_{vp}{}^{n+1} = \epsilon_{vp}{}^n + \Delta\epsilon_{vp}{}^n. \tag{8.27}$$

Arrival at stationary or steady state conditions can be monitored by examination of the strain rates. In particular $\dot{\epsilon}_{vp}$, as given by (8.7), is calculated at each time interval and the time marching process halted as soon as this quantity becomes tolerably small.

8.2.5 Equilibrium correction

The stress increment calculation is based on a linearised form of the incremental equilibrium equations (8.22). Therefore the total stresses, σ^{n+1}, obtained by accumulating all such stress increments are not strictly correct and will not exactly satisfy the equations of equilibrium, (8.21). There are several solution procedures available for applying the necessary correction and Reference 7 discusses the relative merits of various options. The simplest approach is to evaluate σ^{n+1} according to (8.20) and (8.25) and then compute the residual, or out-of-balance, forces, ψ, as

$$\psi^{n+1} = \int_\Omega [B^{n+1}]^T \sigma^{n+1} d\Omega + f^{n+1} \neq 0, \tag{8.28}$$

noting, for geometrically nonlinear problems, that B^{n+1} is evaluated for a displacement state d^{n+1}. This residual force is then added to the applied force increment at the next time step. Such a technique avoids an iteration process and at the same time achieves a reduction in error.

8.3 Selection of the time step length

It can be shown[14] that the time integration scheme formally represented by (8.10) is *unconditionally stable* for values of $\Theta \geqslant \frac{1}{2}$. This implies that the time marching scheme is *numerically* stable but does not guarantee the *accuracy* of the solution at any stage; so that in practice even for values of $\Theta \geqslant \frac{1}{2}$ limits must be placed on the time step length in order to achieve a valid solution.

For $\Theta < \frac{1}{2}$ the integration process is only *conditionally stable* and numerical time integration can only proceed for values of Δt_n less than some critical value. We now proceed to establish rules for choosing the time step length for computation.

Schemes can be employed in which the time step length can be either constant or vary for each time interval. In the variable scheme the magnitude of the time step is controlled by a factor τ which limits the maximum effective viscoplastic strain increment, $\Delta\bar{\epsilon}_{vp}{}^n$ as a fraction of the total effective strain, $\bar{\epsilon}^n$, so that

$$\Delta\bar{\epsilon}_{vp}{}^n = (\sqrt{\tfrac{2}{3}})\{(\dot{\epsilon}_{ij}{}^n)_{vp}(\dot{\epsilon}_{ij}{}^n)_{vp}\}^{1/2}\Delta t_n \leqslant \tau\bar{\epsilon}^n. \tag{8.29}$$

For isoparametric elements, all strains are evaluated at the Gaussian integration points. Therefore Δt_n must be computed to satisfy (8.29) at each such point and the least value taken for analysis. A variant on the above is to limit the time step length according to

$$\{\dot{\epsilon}_{ii}{}^n\}^{\frac{1}{2}}{}_{vp}\Delta t_n \leqslant \tau\{\epsilon_{ii}{}^n\}^{\frac{1}{2}}, \tag{8.30}$$

in which $\epsilon_{ii}{}^n$ is the first total strain invariant and $(\dot{\epsilon}_{ii}{}^n)_{vp}$ is the first viscoplastic strain rate invariant. Thus Δt_n can be formally written for this case as

$$\Delta t_n \leqslant \tau[\epsilon_{ii}{}^n/(\dot{\epsilon}_{ii}{}^n)_{vp}]^{\frac{1}{2}}{}_{\min}. \tag{8.31}$$

The minimum in (8.31) is that taken over all integrating points in the solid. The value of the time increment parameter τ must be specified by the user and for explicit time marching schemes accurate results have been obtained[4,8] in the range $0.01<\tau<0.15$. For implicit schemes, values of τ up to 10 have been found to be stable though the accuracy deteriorates.

Another useful limit can be imposed while using the variable time stepping scheme. The change in the time step length between any two intervals is limited according to

$$\Delta t_{n+1} \leqslant k\Delta t_n, \tag{8.32}$$

where k is a specified constant. Experience suggests a value of $k = 1\cdot5$ to be suitable although there are no fixed criteria for its specification.

The above time step limiting values are basically empirical. Theoretical restrictions on the time step length have been provided by Cormeau[9] for specific forms of the viscoplastic flow rule and for explicit time integration only. In particular, for associated viscoplasticity $Q \equiv F$ and a linear function $\Phi(F) = F$ we have the following limits on the time step length.

$$\Delta t \leqslant \frac{(1+\nu)F_0}{\gamma E} \quad \text{for Tresca materials}$$

$$\Delta t \leqslant \frac{4(1+\nu)F_0}{3\gamma E} \quad \text{Von Mises}$$

$$\Delta t \leqslant \frac{4(1+\nu)(1-2\nu)F_0}{\gamma(1-2\nu+\sin^2\phi)E} \quad \text{Mohr–Coulomb,} \tag{8.33}$$

where γ is the fluidity parameter and ϕ is the angle of internal friction. The term F_0 is the uniaxial yield stress for Tresca and Von Mises solids and is the equivalent value ($c \cos\phi$) for Mohr–Coulomb materials where c is the cohesion. No simple expression exists for the limiting time step length in Drucker–Prager solids.

8.4 Computational procedure

The essential steps in the solution process can be summarised as follows. Solution to the problem must begin from the known initial conditions at time $t = 0$, which are, of course, the solution of the static elastic situation. At this stage $\boldsymbol{d}^0$, $\boldsymbol{F}^0$, $\boldsymbol{\epsilon}^0$, $\boldsymbol{\sigma}^0$ are known and $\boldsymbol{\epsilon}_{vp}{}^0 = \boldsymbol{0}$. The time marching scheme described in Section 8.2.4 can then be employed to advance the solution by one timestep at a time. The solution sequence adopted is as follows.

Stage 1 Suppose at time $t = t_n$ we have an equilibrium situation and $\boldsymbol{d}^n$, $\boldsymbol{\sigma}^n$, $\boldsymbol{\epsilon}^n$, $\boldsymbol{\epsilon}_{vp}{}^n$, $\boldsymbol{F}^n$ are known. The following quantities are assembled:

(a) $$\boldsymbol{B}^n = \boldsymbol{B}_0 + \boldsymbol{B}_{NL}(\boldsymbol{d}^n),$$

(b) $$\boldsymbol{C}^n = \boldsymbol{C}^n(\boldsymbol{\sigma}^n, \Delta t_n),$$

(c) $$\hat{\boldsymbol{D}}^n = (\boldsymbol{D}^{-1} + \boldsymbol{C}^n)^{-1},$$

(d) $$\boldsymbol{K}_T{}^n = \int_\Omega [\boldsymbol{B}^n]^T \hat{\boldsymbol{D}}^n \boldsymbol{B}^n \, d\Omega,$$

(e) $$\dot{\boldsymbol{\epsilon}}_{vp}{}^n = \gamma\langle\Phi\rangle \boldsymbol{a}^n.$$

Stage 2 i) Compute the displacement increments $\Delta\boldsymbol{d}^n$ according to (8.23) as

$$\Delta\boldsymbol{d}^n = [\boldsymbol{K}_T{}^n]^{-1} \Delta\boldsymbol{V}^n,$$

where

$$\Delta\boldsymbol{V}^n = \int_\Omega [\boldsymbol{B}^n]^T \hat{\boldsymbol{D}}^n \dot{\boldsymbol{\epsilon}}_{vp}{}^n \Delta t_n \, d\Omega + \Delta\boldsymbol{f}^n.$$

ii) Calculate the stress increment $\Delta\boldsymbol{\sigma}^n$ as

$$\Delta\boldsymbol{\sigma}^n = \hat{\boldsymbol{D}}^n(\boldsymbol{B}^n \Delta\boldsymbol{d}^n - \dot{\boldsymbol{\epsilon}}_{vp}{}^n \Delta t_n).$$

Stage 3 Determine the total displacements and stresses

$$\boldsymbol{d}^{n+1} = \boldsymbol{d}^n + \Delta\boldsymbol{d}^n$$
$$\boldsymbol{\sigma}^{n+1} = \boldsymbol{\sigma}^n + \Delta\boldsymbol{\sigma}^n.$$

Stage 4 Calculate the viscoplastic strain rate

$$\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1} = \gamma\langle\Phi\rangle \boldsymbol{a}^{n+1}.$$

Stage 5 Apply the equilibrium correction. First calculate $\boldsymbol{B}^{n+1}$ using displacements $\boldsymbol{d}^{n+1}$. Substitute stresses $\boldsymbol{\sigma}^{n+1}$ into the equilibrium equations and evaluate the residual forces $\boldsymbol{\psi}^{n+1}$ as

$$\boldsymbol{\psi}^{n+1} = \int_{\Omega} [\boldsymbol{B}^{n+1}]^T \boldsymbol{\sigma}^{n+1} d\Omega + \boldsymbol{f}^{n+1}.$$

Add these to the vector of incremental pseudo loads for use in the next time step

$$\Delta \boldsymbol{V}^{n+1} = \int_{\Omega} [\boldsymbol{B}^{n+1}]^T \hat{\boldsymbol{D}}^{n+1} \dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1} \Delta t_{n+1} d\Omega + \Delta \boldsymbol{f}^{n+1} + \boldsymbol{\psi}^{n+1}. \tag{8.34}$$

Stage 6 Check to see if the viscoplastic strain rate $\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1}$ is acceptably close to zero at each Gaussian integrating point throughout the structure (i.e. to within a specified tolerance).

If so, steady state conditions are deemed to have been achieved and the solution is either terminated or the next load increment is applied. If $\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1}$ is non-zero return to Stage 1 and repeat the entire procedure for the next time step.

The above algorithm can be employed with either a constant or variable time step length. For the variable time step option the interval length Δt_{n+1}, for the next time step must be calculated according to (8.29) or (8.31) subject to the restriction of (8.32).

8.5 Evaluation of matrix, *H*

For solution by the fully implicit or semi-implicit (trapezoidal) time stepping scheme, matrix $\boldsymbol{C}^n$ is required which in turn can be expressed in terms of $\boldsymbol{H}^n$ as indicated in (8.14). Matrix $\boldsymbol{H}^n$ must be explicitly determined for the yield criterion assumed for material behaviour. From (8.7) and (8.12) we have

$$\boldsymbol{H} = \frac{\partial \dot{\boldsymbol{\epsilon}}_{vp}}{\partial \boldsymbol{\sigma}^n} = \gamma \left\{ \Phi \frac{\partial \boldsymbol{a}^T}{\partial \boldsymbol{\sigma}} + \frac{d\Phi}{dF} \boldsymbol{a}\, \boldsymbol{a}^T \right\}, \tag{8.35}$$

where the symbols $\langle\ \rangle$ on Φ and the superscript n are dropped for convenience. Restricting discussion to the *Von Mises* yield criterion we have, from (7.64),

$$\boldsymbol{a}^T = \frac{\partial F}{\partial \boldsymbol{\sigma}} = \frac{\partial[(\sqrt{3})(J_2{}')^{1/2}]}{\partial \boldsymbol{\sigma}}, \tag{8.36}$$

or

$$\boldsymbol{a}^T = \frac{\partial F}{\partial J_2{}'} \frac{\partial J_2{}'}{\partial \boldsymbol{\sigma}} = \frac{\sqrt{3}}{2(J_2{}')^{1/2}} \{\sigma_x{}', \sigma_y{}', \sigma_z{}', 2\tau_{yz}, 2\tau_{zx}, 2\tau_{xy}\}, \tag{8.37}$$

for a three dimensional situation. Thus

$$\boldsymbol{a}\boldsymbol{a}^T = \frac{3}{4J_2'}\boldsymbol{M}_2, \tag{8.38}$$

where

$$\boldsymbol{M}_2 = \begin{bmatrix} (\sigma_x')^2 & \sigma_x'\sigma_y' & \sigma_x'\sigma_z' & 2\sigma_x'\tau_{yz} & 2\sigma_x'\tau_{zx} & 2\sigma_x'\tau_{xy} \\ & (\sigma_y')^2 & \sigma_y'\sigma_z' & 2\sigma_y'\tau_{yz} & 2\sigma_y'\tau_{zx} & 2\sigma_y'\tau_{xy} \\ & & (\sigma_z')^2 & 2\sigma_z'\tau_{yz} & 2\sigma_z'\tau_{zx} & 2\sigma_z'\tau_{xy} \\ & & & 4(\tau_{yz})^2 & 4\tau_{yz}\tau_{zx} & 4\tau_{yz}\tau_{xy} \\ & \textit{Symmetric} & & & 4(\tau_{zx})^2 & 4\tau_{zx}\tau_{xy} \\ & & & & & 4(\tau_{xy})^2 \end{bmatrix}. \tag{8.39}$$

Also from (8.37)

$$\frac{\partial \boldsymbol{a}^T}{\partial \boldsymbol{\sigma}} = \frac{\sqrt{3}}{2(J_2')^{1/2}}\boldsymbol{M}_1 - \frac{\sqrt{3}}{4(J_2')^{3/2}}\boldsymbol{M}_2, \tag{8.40}$$

where

$$\boldsymbol{M}_1 = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & 0 & 0 & 0 \\ & \frac{2}{3} & -\frac{1}{3} & 0 & 0 & 0 \\ & & \frac{2}{3} & 0 & 0 & 0 \\ & & & 2 & 0 & 0 \\ & & \textit{Symmetric} & & 2 & 0 \\ & & & & & 2 \end{bmatrix}. \tag{8.41}$$

Substituting from (8.38) and (8.40) into (8.35), and restoring the symbols $\langle\ \rangle$, we have finally

$$\boldsymbol{H} = p_1\boldsymbol{M}_1 + p_2\boldsymbol{M}_2, \tag{8.42}$$

where

$$p_1 = \gamma\left\langle\frac{\sqrt{3}}{2(J_2')^{1/2}}.\Phi\right\rangle$$

$$p_2 = \gamma\left\langle\frac{3}{4J_2'}\,\frac{d\Phi}{dF} - \frac{(\sqrt{3})\Phi}{4(J_2')^{3/2}}\right\rangle. \tag{8.43}$$

The form of $d\Phi/dF$ depends on the explicit form of Φ employed, examples of which were given in (8.8) and (8.9). Matrix $\boldsymbol{H}^n$ is then obtained by using stresses $\boldsymbol{\sigma}^n$ to evaluate J_2' and $\boldsymbol{M}_2$.

For two-dimensional situations (plane stress, plane strain and axial symmetry) the only relevant stress terms are given in (7.72). In this case $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ reduce, on deletion of the appropriate terms, to

$$\boldsymbol{M}_1 = \left[\begin{array}{ccc|c} \frac{2}{3} & -\frac{1}{3} & 0 & -\frac{1}{3} \\ & \frac{2}{3} & 0 & -\frac{1}{3} \\ \multicolumn{2}{c}{\text{Symmetric}} & 2 & 0 \\ \hline & & & \frac{2}{3} \end{array}\right], \tag{8.44}$$

and

$$\boldsymbol{M}_2 = \left[\begin{array}{ccc|c} (\sigma_x')^2 & \sigma_x'\sigma_y' & 2\sigma_x'\tau_{xy} & \sigma_x'\sigma_z' \\ & (\sigma_y')^2 & 2\sigma_y'\tau_{xy} & \sigma_y'\sigma_z' \\ \multicolumn{2}{c}{\text{Symmetric}} & 4(\tau_{xy})^2 & 2\tau_{xy}\sigma_z' \\ \hline & & & (\sigma_z')^2 \end{array}\right], \tag{8.45}$$

and J_2' is given by (7.76). For plane stress and plane strain problems only the upper 3×3 partition is employed while for axisymmetric situations the complete matrices are utilised with x and y being replaced by r and z respectively.

Similar expressions can be derived for the Tresca, Mohr–Coulomb and Drucker–Prager yield criteria by employing the appropriate expression for F in (8.36) and repeating the above calculations. The form of F is given in (7.63), (7.65) and (7.66) for the Tresca, Mohr–Coulomb and Drucker–Prager laws respectively.

8.6 Program structure

The computation sequence for the program is shown in Fig. 8.1. The program structure follows closely that for static elasto-plastic analysis described in Chapter 7. In fact, the majority of the subroutines utilised are common to both applications and it is only the additional subroutines required that are described in this chapter. For the viscoplastic program the time stepping loop replaces the nonlinear solution iteration loop for conventional plasticity and subroutine STEPVP, whose main role is to evaluate quantities at the end of a timestep, replaces the plasticity subroutine RESIDU. In this chapter we need to describe in detail subroutines STIFVP, TANGVP, STEPVP, FLOWVP and STEADY. The descriptions of all other subroutines required for assembly of a working viscoplastic program have been given in Chapters 6 and 7. The version described is restricted to the case of infinitesimal strains. The modifications required to include large deformation effects are straightforward and are left as an exercise to the reader. Furthermore, for implicit schemes, only the Von Mises yield criterion is considered.

The list of material properties accepted in subroutine INPUT described in Section 6.5.1 must be extended beyond those required for elasto-plastic analysis, since additional material parameters are required to define the

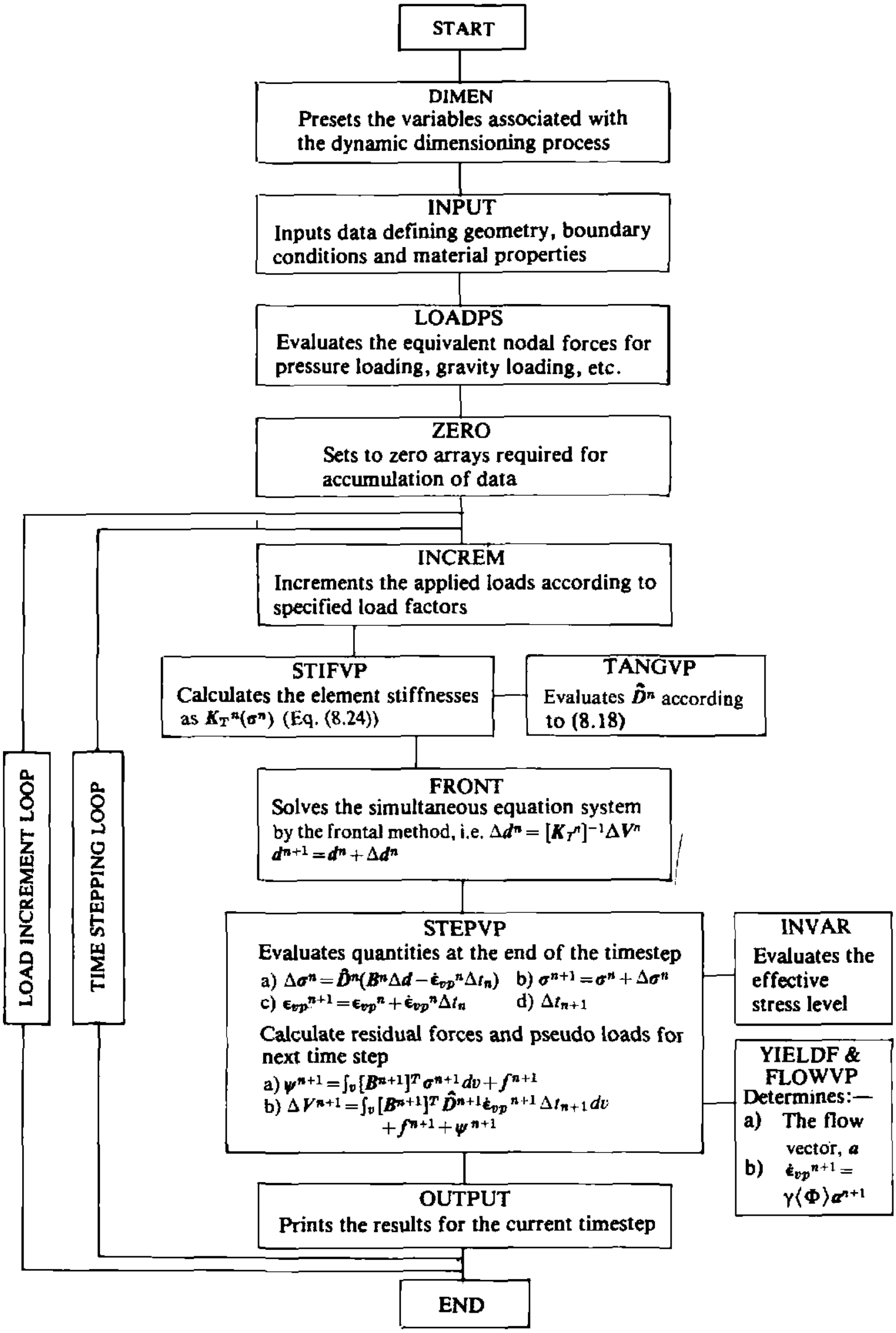


Fig. 8.1 Flow sequence for the two-dimensional elasto-viscoplastic stress analysis program.

viscoplastic flow. This is accomplished by specifying the value of NPROP as 10 in subroutine DIMEN, described in Section 7.8.1, and inputting the following properties for each different material.

PROPS(NUMAT, 1)	Elastic modulus, E.
PROPS(NUMAT, 2)	Poissons ratio, ν.
PROPS(NUMAT, 3)	Material thickness, t.
PROPS(NUMAT, 4)	Material mass density, ρ.
PROPS(NUMAT, 5)	Uniaxial yield stress σ_Y (Tresca and Von Mises solids); Cohesion c (Mohr–Coulomb and Drucker–Prager materials).
PROPS(NUMAT, 6)	Hardening parameter H' for linear strain hardening.
PROPS(NUMAT, 7)	Angle of internal friction for Mohr–Coulomb and Drucker–Prager materials only.
PROPS(NUMAT, 8)	The fluidity parameter, γ.
PROPS(NUMAT, 9)	The coefficient M in (8.8) or coefficient N in (8.9).
PROPS(NUMAT, 10)	Indicator specifying type of flow function to be employed: 0 – Flow function (8.8) 1 – Flow function (8.9)

8.7 Formulation of the tangential stiffness matrix

The role of the subroutines described in this section is to calculate the tangential stiffness matrix for each element according to (8.24). The complete operation is shared between three subroutines which will now be described.

8.7.1 Subroutine STIFVP

This subroutine controls the overall formulation of the tangential stiffness matrix for each element and is very similar to subroutine STIFFP, described in Section 7.8.5, which performs the same task for conventional plasticity. For the case of small deformations, matrix $\boldsymbol{B}^n$ is constant and equal to $\boldsymbol{B}_0$ the usual infinitesimal elastic value. Matrix $\boldsymbol{B}_0$ is given by subroutine BMATPS described in Section 6.4.7. To evaluate $\boldsymbol{K}_T{}^n$ it is necessary to find $\hat{\boldsymbol{D}}^n$ whose precise form is given by (8.18). With the normal elastic material matrix $\boldsymbol{D}$ replaced by $\hat{\boldsymbol{D}}^n$, the stiffness evaluation follows the standard procedure described in Section 7.8.5. Subroutine STIFVP can now be presented and described.

```
      SUBROUTINE STIFVP(COORD,IINCS,LNODS,MATNO,MEVAB,MMATS,
     .                  MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE,NSTRE,
     .                  NSTR1,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     .                  STRSG,NTYPE,NCRIT,TIMEX,DTIME)
C*********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE STIFFNESS MATRIX FOR EACH ELEMENT
C     IN TURN
C
C*********************************************************************
      DIMENSION BMATX(4,18),CARTD(2,9),COORD(MPOIN,2),DBMAT(4,18),
     .          DERIV(2,9),DEVIA(4),DMATX(4,4),
     .          ELCOD(2,9),EPSTN(MTOTG),ESTIF(18,18),LNODS(MELEM,9),
     .          MATNO(MELEM),POSGP(4),PROPS(MMATS,10),SHAPE(9),
     .          WEIGP(4),STRES(4),STRSG(4,MTOTG),
```

```
     .            DVECT(4),AVECT(4),GPCOD(2,9)
      TWOPI=6.283185308
      REWIND 1
      KGAUS=0
C
C*** LOOP OVER EACH ELEMENT
C
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 10 INODE=1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      IPOSN=(LNODE-1)*2
      DO 10 IDIME=1,2
      IPOSN=IPOSN+1
   10 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
      THICK=PROPS(LPROP,3)
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
C
      DO 20 IEVAB=1,NEVAB
      DO 20 JEVAB=1,NEVAB
   20 ESTIF(IEVAB,JEVAB)=0.0
      KGASP=0
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      DO 50 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      DO 50 JGAUS=1,NGAUS
      ETASP=POSGP(JGAUS)
      KGASP=KGASP+1
      KGAUS=KGAUS+1
C
C*** EVALUATE THE D-MATRIX
C
      CALL MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL VOLUME,ETC.
C
      CALL       SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL       JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .                  NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(THICK.NE.0.0) DVOLU=DVOLU*THICK
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      DO 25 ISTR1=1,NSTR1
   25 STRES(ISTR1)=STRSG(ISTR1,KGAUS)
      IF(TIMEX.GT.0.0) CALL TANGVP(LPROP,STRES,PROPS,TIMEX,DTIME,
     .                         NSTRE,NTYPE,MMATS,NCRIT,DMATX)
      CALL      DBE(BMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTRE,NSTR1)
C
C*** CALCULATE THE ELEMENT STIFFNESSES
C
      DO 30 IEVAB=1,NEVAB
      DO 30 JEVAB=IEVAB,NEVAB
      DO 30 ISTRE=1,NSTRE
   30 ESTIF(IEVAB,JEVAB)=ESTIF(IEVAB,JEVAB)+BMATX(ISTRE,IEVAB)*
     . DBMAT(ISTRE,JEVAB)*DVOLU
```

```
   50 CONTINUE                                                          STVP  81
C                                                                       STVP  82
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX               STVP  83
C                                                                       STVP  84
      DO 60 IEVAB=1,NEVAB                                               STVP  85
      DO 60 JEVAB=1,NEVAB                                               STVP  86
   60 ESTIF(JEVAB,IEVAB)=ESTIF(IEVAB,JEVAB)                             STVP  87
C                                                                       STVP  88
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX AND SAMPLING POINT        STVP  89
C    COORDINATES FOR EACH ELEMENT ON DISC FILE                          STVP  90
C                                                                       STVP  91
      WRITE(1) ESTIF                                                    STVP  92
   70 CONTINUE                                                          STVP  93
      RETURN                                                            STVP  94
      END                                                               STVP  95
```

STVP 17 Compute the value of 2π.

STVP 18 Rewind the disc file on which the element stiffness matrices will be stored in turn.

STVP 19 Set to zero the counter which indicates the *overall* Gauss point location.

STVP 23 Enter the loop over each element in the structure.

STVP 24 Identify the material property type of the current element.

STVP 28–33 Store the element nodal coordinates in the local array ELCOD for convenient use later.

STVP 34 Identify the element thickness.

STVP 38–40 Zero the element stiffness array.

STVP 41 Set to zero the *element* Gauss point counter.

STVP 45–48 Enter the numerical integration loops and locate the position (ξ, η) of the current point.

STVP 49–50 Increment the local and global Gauss point counters.

STVP 54 Call subroutine MODPS to evaluate the elasticity matrix, $\boldsymbol{D}$.

STVP 58 Evaluate the shape functions N_i and $\partial N_i/\partial\xi$, $\partial N_i/\partial\eta$ for the current Gauss point.

STVP 59–60 Evaluate the Gauss point coordinates, GPCOD(IDIME, KGASP), the determinant of the Jacobian matrix $|\boldsymbol{J}|$ and the Cartesian derivatives of the shape functions $\partial N_i/\partial x$, $\partial N_i/\partial y$ (or $\partial N_i/\partial r$, $\partial N_i/\partial z$ for axisymmetric problems).

STVP 61–63 Calculate the elemental volume for numerical integration as $|\boldsymbol{J}|\,W_\xi W_\eta$ taking care to multiply by the appropriate element thickness or by $2\pi r$ for axisymmetric problems.

STVP 67 Evaluate the $\boldsymbol{B}$ matrix.

STVP 68–69 Store the current stresses in a local array.

STVP 70–71 For an implicit or semi-implicit timestepping scheme ($\Theta \neq 0$), call subroutine TANGVP to evaluate $\hat{\boldsymbol{D}}^n$ which is stored as DMATX.

STVP 72 Evaluate $\boldsymbol{DB}$ (or $\hat{\boldsymbol{D}}^n\boldsymbol{B}$ for implicit schemes).

STVP 76–80 Compute the upper triangle of the element stiffness matrix as

$$\int_{\Omega} \boldsymbol{B}^T \hat{\boldsymbol{D}}^n \boldsymbol{B} \, d\Omega.$$

STVP 81 End of loop for numerical integration.

STVP 85–87 Complete the lower triangle of the element stiffness matrix by symmetry.

STVP 92 Store the element stiffness matrix on disc file 1.

STVP 93 Return to process the next element.

8.7.2 Subroutine TANGVP

The function of this subroutine is to evaluate $\hat{\boldsymbol{D}}^n$ for use in (8.24). Matrix $\hat{\boldsymbol{D}}^n$, which is defined in (8.18), is stress dependent and therefore must be calculated for each Gaussian integrating point in turn. The computational sequence followed is:

a) Evaluate $\boldsymbol{H}^n$ according to (8.42)
b) Calculate $\boldsymbol{C}^n$ according to (8.14)
c) Evaluate $\hat{\boldsymbol{D}}^n$ according to (8.18)

Two forms of the flow function Φ are considered as defined in (8.8) and (8.9). Thus, for use in (8.43), we have

$$\frac{d\Phi}{dF} = \frac{M}{F_0} e^{M\left(\frac{F-F_0}{F_0}\right)}$$

or

$$\frac{d\Phi}{dF} = \frac{N}{F_0}\left(\frac{F-F_0}{F_0}\right)^{N-1}. \tag{8.46}$$

Array DMATX which originally contains the elastic matrix $\boldsymbol{D}$ is used to finally store $\hat{\boldsymbol{D}}^n$. The matrix inversions required in (8.18) are performed by a separate subroutine, INVERT.

Subroutine TANGVP is now presented and described.

```
      SUBROUTINE TANGVP(LPROP,STRES,PROPS,TIMEX,DTIME,
     .                  NSTRE,NTYPE,MMATS,NCRIT,DMATX)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE PSEUDO D-MATRIX
C
C***********************************************************************
      DIMENSION STRES(4),CMATX(4,4),TMATX(4,4),TRIX1(4,4),TRIX2(4,4),
     .          PROPS(MMATS,10),DEVIA(4),DMATX(4,4)
      ROOT3=1.73205080757
      FDATM=PROPS(LPROP,5)
      GAMMA=PROPS(LPROP,8)
      DELTA=PROPS(LPROP,9)
      NFLOW=PROPS(LPROP,10)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,THETA,
     .          VARJ2,YIELD)
      FCURR=YIELD-FDATM
      FNORM=FCURR/FDATM
      IF(FNORM.LE.0.0) RETURN
      IF(NFLOW.EQ.1) GO TO 10
      CMULT=EXP(DELTA*FNORM)-1.0
      GRADP=DELTA*(EXP(DELTA*FNORM))/FDATM
      GO TO 20
```

```
   10 CMULT=FNORM**DELTA                                                    TGVP  24
      GRADP=DELTA*(FNORM**(DELTA-1.0))/FDATM                                TGVP  25
   20 FACT1=GAMMA*ROOT3*CMULT/(2.0*STEFF)                                   TGVP  26
      FACT2=GAMMA*(0.75*GRADP/VARJ2-3.0*CMULT/(4.0*ROOT3*STEFF*VARJ2))      TGVP  27
C                                                                           TGVP  28
C*** MATRICES M1 AND M2 FOR A VON MISES MATERIAL                            TGVP  29
C                                                                           TGVP  30
      TRIX1(1,1)=0.666666667                                                TGVP  31
      TRIX1(1,2)=-0.333333333                                               TGVP  32
      TRIX1(1,3)=0.0                                                        TGVP  33
      TRIX1(2,2)=0.666666667                                                TGVP  34
      TRIX1(2,3)=0.0                                                        TGVP  35
      TRIX1(3,3)=2.0                                                        TGVP  36
      IF(NTYPE.NE.3) GO TO 30                                               TGVP  37
      TRIX1(1,4)=-0.333333333                                               TGVP  38
      TRIX1(2,4)=-0.333333333                                               TGVP  39
      TRIX1(3,4)=0.0                                                        TGVP  40
      TRIX1(4,4)=0.666666667                                                TGVP  41
   30 TRIX2(1,1)=DEVIA(1)*DEVIA(1)                                          TGVP  42
      TRIX2(1,2)=DEVIA(1)*DEVIA(2)                                          TGVP  43
      TRIX2(1,3)=2.0*DEVIA(1)*DEVIA(3)                                      TGVP  44
      TRIX2(2,2)=DEVIA(2)*DEVIA(2)                                          TGVP  45
      TRIX2(2,3)=2.0*DEVIA(2)*DEVIA(3)                                      TGVP  46
      TRIX2(3,3)=4.0*DEVIA(3)*DEVIA(3)                                      TGVP  47
      IF(NTYPE.NE.3) GO TO 40                                               TGVP  48
      TRIX2(1,4)=DEVIA(1)*DEVIA(4)                                          TGVP  49
      TRIX2(2,4)=DEVIA(2)*DEVIA(4)                                          TGVP  50
      TRIX2(3,4)=2.0*DEVIA(3)*DEVIA(4)                                      TGVP  51
      TRIX2(4,4)=DEVIA(4)*DEVIA(4)                                          TGVP  52
   40 DO 50 ISTRE=1,NSTRE                                                   TGVP  53
      DO 50 JSTRE=1,NSTRE                                                   TGVP  54
      TRIX1(JSTRE,ISTRE)=TRIX1(ISTRE,JSTRE)                                 TGVP  55
   50 TRIX2(JSTRE,ISTRE)=TRIX2(ISTRE,JSTRE)                                 TGVP  56
      DO 60 ISTRE=1,NSTRE                                                   TGVP  57
      DO 60 JSTRE=1,NSTRE                                                   TGVP  58
   60 CMATX(ISTRE,JSTRE)=TIMEX*DTIME*(FACT1*TRIX1(ISTRE,JSTRE)              TGVP  59
     . +FACT2*TRIX2(ISTRE,JSTRE))                                           TGVP  60
      CALL INVERT(DMATX,TMATX,NSTRE)                                        TGVP  61
      DO 70 ISTRE=1,NSTRE                                                   TGVP  62
      DO 70 JSTRE=1,NSTRE                                                   TGVP  63
   70 TMATX(ISTRE,JSTRE)=TMATX(ISTRE,JSTRE)+CMATX(ISTRE,JSTRE)              TGVP  64
      CALL INVERT(TMATX,DMATX,NSTRE)                                        TGVP  65
      RETURN                                                                TGVP  66
      END                                                                   TGVP  67
```

TGVP 10 Evaluate $\sqrt{(3)}$.

TGVP 11 Identify the yield stress F as FDATM.

TGVP 12 Identify the fluidity parameter γ as GAMMA.

TGVP 13 For flow law (8.8) store the index M as DELTA, or for flow law (8.9) store the index N as DELTA.

TGVP 14 Identify the type of flow function to be used as governed by material property PROPS(LPROP,10) supplied as input:
NFLOW = 0 – Flow function (8.8) to be used,
NFLOW = 1 – Flow function (8.9) to be used.

TGVP 15–16 Call subroutine INVAR to evaluate the effective stress components, the effective stress level and J_2'.

TGVP 17–18 Evaluate $F\text{–}F_0/F_0$ as FNORM.

TGVP 21–22	Evaluate Φ and $d\Phi/dF$ for flow function (8.8).
TGVP 24–25	Evaluate Φ and $d\Phi/dF$ for flow function (8.9).
TGVP 26–27	Compute p_1 and p_2 according to (8.43).
TGVP 31–41	Evaluate $\boldsymbol{M}_1$ according to (8.44) taking the full 4×4 matrix for axisymmetric situations.
TGVP 42–52	Evaluate $\boldsymbol{M}_2$ according to (8.45) taking the full 4×4 matrix for axisymmetric situations.
TGVP 53–56	Complete the lower triangle of $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ by symmetry.
TGVP 57–60	Compute matrix $\boldsymbol{C}^n$ according to (8.14) and (8.42).
TGVP 61	Call subroutine INVERT to evaluate $\boldsymbol{D}^{-1}$ and store as TMATX.
TGVP 62–64	Compute $\boldsymbol{D}^{-1}+\boldsymbol{C}^n$.
TGVP 65	Call subroutine INVERT to evaluate $(\boldsymbol{D}^{-1}+\boldsymbol{C}^n)^{-1}$ and store as DMATX.

8.7.3 Subroutine INVERT

The function of this subroutine is to determine the inverse of any arbitrary square matrix. In particular, the subroutine accepts a matrix AMATX with dimensions NARAY×NARAY and evaluates the inverse as BMATX. The procedure employed is the standard method of reduction in which starting from the original matrix AMATX and assuming an identity matrix for BMATX, an elimination process is followed until AMATX is reduced to an identity form. Then at this stage BMATX is the inverse of AMATX.

The subroutine is presented below without further comment.

```
      SUBROUTINE INVERT(AMATX,BMATX,NARAY)
C**********************************************************************
C
C*** TO PROVIDE THE INVERSE OF AMATX AS BMATX
C
C**********************************************************************
      DIMENSION AMATX(4,4),BMATX(4,4)
      DO 10 IARAY=1,NARAY
      DO 10 JARAY=1,NARAY
      BMATX(IARAY,JARAY)=0.0
   10 IF(IARAY.EQ.JARAY) BMATX(IARAY,JARAY)=1.0
      DO 20 IARAY=1,NARAY
      DENOM=AMATX(IARAY,IARAY)
      DO 30 JARAY=1,NARAY
      AMATX(IARAY,JARAY)=AMATX(IARAY,JARAY)/DENOM
   30 BMATX(IARAY,JARAY)=BMATX(IARAY,JARAY)/DENOM
      KARAY=IARAY+1
      IF(KARAY.GT.NARAY) GO TO 40
      DO 20 JARAY=KARAY,NARAY
      CONST=AMATX(JARAY,IARAY)
      DO 20 LARAY=IARAY,NARAY
      AMATX(JARAY,LARAY)=AMATX(JARAY,LARAY)-AMATX(IARAY,LARAY)
     . *CONST
   20 BMATX(JARAY,LARAY)=BMATX(JARAY,LARAY)-BMATX(IARAY,LARAY)
     . *CONST
   40 CONTINUE
      DO 50 IARAY=2,NARAY
      KARAY=NARAY-IARAY+2
```

```
      LIMIT=KARAY-1
      DO 50 LARAY=1,LIMIT
      CONST=AMATX(LARAY,KARAY)
      DO 50 JARAY=1,KARAY
      AMATX(LARAY,JARAY)=AMATX(LARAY,JARAY)-AMATX(KARAY,JARAY)
     . *CONST
   50 BMATX(LARAY,JARAY)=BMATX(LARAY,JARAY)-BMATX(KARAY,JARAY)
     . *CONST
      RETURN
      END
```

8.8 Subroutine STEPVP for the evaluation of end of time step quantities and equilibrium correction terms

With reference to Fig. 8.1, this subroutine evaluates quantities, such as stresses and viscoplastic strains, at the end of the current timestep and also calculates the loading to be applied during the next timestep. The subroutine is structured to perform the following operations sequentially:

(a) All quantities at the end of timestep n are calculated as $(\ )^{n+1}$.

(b) Subroutine INVAR, YIELDF and FLOWVP are called to evaluate the current viscoplastic flow rate, $\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1}$.

(c) The maximum permissible interval length, Δt_{n+1}, for the next timestep as governed by (8.29) and (8.32) is calculated.

(d) The residual forces, ψ^{n+1}, are evaluated and the loads, ΔV^{n+1}, for the next timestep then calculated.

In the program presented we restrict ourselves to loads applied in discrete increments. An increment of load is applied and the time stepping process is followed until either steady state conditions are achieved, or a specified number of timesteps is reached. Then a further increment of load is applied and the process repeated. Thus in (8.23), $\Delta f^n = \mathbf{0}$ for all stages other than the first timestep of a particular load increment.

The attainment of steady state conditions can be monitored by accumulating some measure of the viscoplastic strain rate for all Gauss points in the structure. At steady state this quantity will become zero. The degree of total viscoplastic flow at any point is best monitored by evaluating the total effective viscoplastic strain rate at all Gauss points according to

$$\bar{\dot{\epsilon}}_{vp} = (\sqrt{\tfrac{2}{3}})\{(\dot{\epsilon}_{ij})_{vp}(\dot{\epsilon}_{ij})_{vp}\}^{1/2}. \tag{8.47}$$

Subroutine STEPVP is now presented and described.

```
      SUBROUTINE STEPVP(ASDIS,COORD,ELOAD,ISTEP,LNODS,LPROP,TIMEX,
     .                  MATNO,MELEM,MMATS,MPOIN,MTOTG,TAUFT,DTIME,
     .                  MTOTV,NDOFN,NELEM,NEVAB,NGAUS,NNODE,NSTR1,
     .                  NTYPE,POSGP,PROPS,NSTRE,NCRIT,STRSG,WEIGP,
     .                  TDISP,VISTN,VIVEL,TLOAD,FTIME,DTINT,IINCS)
C*********************************************************************
C
C**** EVALUATES QUANTITIES AT END OF TIME STEP AND CALCULATES THE
C     RESIDUAL FORCES AND PSEUDO FORCES FOR THE NEXT STEP
C
C*********************************************************************
```

```
      DIMENSION ASDIS(MTOTV),AVECT(4),CARTD(2,9),COORD(MPOIN,2),
     .          DEVIA(4),ELCOD(2,9),ELDIS(2,9),ELOAD(MELEM,18),
     .          LNODS(MELEM,9),POSGP(4),PROPS(MMATS,10),STRAN(4),
     .          STRES(4),STRSG(4,MTOTG),VIVEL(5,MTOTG),
     .          VISTN(4,MTOTG),WEIGP(4),DMATX(4,4),TLDIS(2,9),
     .          DERIV(2,9),SHAPE(9),GPCOD(2,9),TDISP(MTOTV),
     .          MATNO(MELEM),DJACM(2,2),BMATX(4,18),DESTN(4),
     .          TLOAD(MELEM,18),SVECT(4)
      TWOPI=6.283185308
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=0.0
      KGAUS=0
      DNEXT=FTIME*DTIME
      DO 80 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** STORE COORDINATES AND INCREMENTAL DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 20 INODE=1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      NPOSN=(LNODE-1)*NDOFN
      DO 20 IDOFN=1,NDOFN
      NPOSN=NPOSN+1
      ELCOD(IDOFN,INODE)=COORD(LNODE,IDOFN)
      TLDIS(IDOFN,INODE)=TDISP(NPOSN)
   20 ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
      THICK=PROPS(LPROP,3)
      KGASP=0
      DO 70 IGAUS=1,NGAUS
      DO 70 JGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      KGAUS=KGAUS+1
      KGASP=KGASP+1
      CALL MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
      DO 30 ISTR1=1,NSTR1
   30 STRES(ISTR1)=STRSG(ISTR1,KGAUS)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,THETA,
     .           VARJ2,YIELD)
      IF(TIMEX.GT.0.0) CALL TANGVP(LPROP,STRES,PROPS,TIMEX,DTIME,
     .                             NSTRE,NTYPE,MMATS,NCRIT,DMATX)
      CALL SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(THICK.NE.0.0) DVOLU=DVOLU*THICK
      CALL STRESS(DMATX,LPROP,NTYPE,PROPS,NDOFN,CARTD,ELDIS,SHAPE,
     .            GPCOD,NSTRE,VIVEL,DTIME,STRSG,KGASP,MTOTG,MMATS,
     .            SVECT,NNODE,NSTR1,KGAUS,TLDIS)
      DO 60 ISTR1=1,NSTR1
      DESTN(ISTR1)=VIVEL(ISTR1,KGAUS)*DTIME
   60 VISTN(ISTR1,KGAUS)=VISTN(ISTR1,KGAUS)+DESTN(ISTR1)
      DEBAR=SQRT((2.0*(DESTN(1)*DESTN(1)+DESTN(2)*DESTN(2)+DESTN(4)*
     . DESTN(4))+DESTN(3)*DESTN(3))/3.0)
      DO 65 ISTR1=1,NSTR1
   65 STRES(ISTR1)=STRSG(ISTR1,KGAUS)
      VIVEL(5,KGAUS)=VIVEL(5,KGAUS)+DEBAR
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,THETA,
     .           VARJ2,YIELD)
      CALL YIELDF(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .            PROPS,SINT3,STEFF,THETA,VARJ2)
      CALL FLOWVP(AVECT,PROPS,LPROP,STEFF,NSTR1,MTOTG,VIVEL,
     .            YIELD,KGAUS,MMATS,NCRIT,FNORM,ALLOW)
```

```
      IF(FNORM.LT.ALLOW) GO TO 70
      EPBAR=SQRT((2.0*(AVECT(1)*AVECT(1)+AVECT(2)*AVECT(2)+AVECT(4)
     . *AVECT(4))+AVECT(3)*AVECT(3))/3.0)
      TSBAR=SQRT((2.0*(SVECT(1)*SVECT(1)+SVECT(2)*SVECT(2)+SVECT(4)
     . *SVECT(4))+SVECT(3)*SVECT(3))/3.0)
      DELTM=TAUFT*TSBAR/EPBAR
      IF(DELTM.LT.DNEXT) DNEXT=DELTM
   70 CONTINUE
   80 CONTINUE
      DTIME=DNEXT
      IF(ISTEP.EQ.1) DTIME=DTINT
      KGAUS=0
      DO 140 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      DO 90 INODE=1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      NPOSN=(LNODE-1)*NDOFN
      DO 90 IDOFN=1,NDOFN
      NPOSN=NPOSN+1
   90 ELCOD(IDOFN,INODE)=COORD(LNODE,IDOFN)
      THICK=PROPS(LPROP,3)
      KGASP=0
      DO 130 IGAUS=1,NGAUS
      DO 130 JGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      KGAUS=KGAUS+1
      KGASP=KGASP+1
      CALL SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(THICK.NE.0.0) DVOLU=DVOLU*THICK
      CALL BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      CALL MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
      DO 100 ISTR1=1,NSTR1
  100 STRES(ISTR1)=STRSG(ISTR1,KGAUS)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,THETA,
     .           VARJ2,YIELD)
      IF(TIMEX.GT.0.0) CALL TANGVP(LPROP,STRES,PROPS,TIMEX,DTIME,
     .                             NSTRE,NTYPE,MMATS,NCRIT,DMATX)
C
C*** CALCULATE THE RESIDUAL FORCES AND INCREMENTAL PSEUDO LOADS
C
      DO 110 ISTRE=1,NSTRE
      STRES(ISTRE)=0.0
      DO 110 JSTRE=1,NSTRE
  110 STRES(ISTRE)=STRES(ISTRE)+DMATX(ISTRE,JSTRE)*VIVEL(JSTRE,KGAUS)
     . *DTIME
      MGASH=0
      DO 120 INODE=1,NNODE
      DO 120 IDOFN=1,NDOFN
      MGASH=MGASH+1
      DO 120 ISTRE=1,NSTRE
  120 ELOAD(IELEM,MGASH)=ELOAD(IELEM,MGASH)+BMATX(ISTRE,MGASH)
     . *(STRES(ISTRE)-STRSG(ISTRE,KGAUS))*DVOLU
  130 CONTINUE
  140 CONTINUE
      DO 150 IELEM=1,NELEM
      DO 150 IEVAB=1,NEVAB
  150 ELOAD(IELEM,IEVAB)=ELOAD(IELEM,IEVAB)+TLOAD(IELEM,IEVAB)
      RETURN
      END
```

SPVP 20	Compute 2π.
SPVP 21–23	Zero the array in which the pseudo loads for the next timestep will be stored.
SPVP 24	Zero the Gauss point counter over all elements.
SPVP 25	Increase the timestep length from the value used for the previous step by the factor FTIME. If this new value is less than that predicted later in this routine, this step length will be employed for the next time step.
SPVP 26	Loop over each element.
SPVP 27	Identify the element material property number.
SPVP 32–39	Store the element coordinates in array ELCOD, the incremental displacements $\Delta \boldsymbol{d}^n$ in ELDIS and the total displacements $\boldsymbol{d}^n$ in TLDIS.
SPVP 40	Identify the element thickness.
SPVP 41	Zero the local Gauss point counter.
SPVP 42–45	Enter the loops for numerical integration and evaluate the local coordinates (ξ, η) at the sampling point.
SPVP 46–47	Increment the local and global Gauss point counters.
SPVP 48	Compute the elasticity matrix, $\boldsymbol{D}$.
SPVP 49–50	Store the total current stresses $\boldsymbol{\sigma}^n$ locally in STRES.
SPVP 51–52	Evaluate the deviatoric stresses and J_2'.
SPVP 53–54	For the implicit or semi-implicit time stepping scheme evaluate $\hat{\boldsymbol{D}}^n$.
SPVP 55	Evaluate the shape functions N_i and the derivatives $\partial N_i/\partial \xi$, $\partial N_i/\partial \eta$.
SPVP 56	Evaluate the Gauss point coordinates GPCOD(IDIME, KGASP), the determinant of the Jacobian matrix $\lvert \boldsymbol{J} \rvert$ and the Cartesian derivatives of the shape functions.
SPVP 57–59	Calculate the elemental volume for numerical integration as $\lvert \boldsymbol{J} \rvert W_\xi W_\eta$ taking care to multiply by $2\pi r$ for axisymmetric problems.
SPVP 60–62	Call subroutine STRESS to evaluate the stress increment $\Delta\boldsymbol{\sigma}^n$ according to (8.20) and also $\boldsymbol{\sigma}^{n+1} = \boldsymbol{\sigma}^n + \Delta\boldsymbol{\sigma}^n$.
SPVP 63–65	Evaluate the incremental viscoplastic strain and the total current viscoplastic strain, $\boldsymbol{\epsilon}_{vp}{}^{n+1}$.
SPVP 66–67	Accumulate the absolute value of the viscoplastic strain increment. This will allow us to monitor whether or not steady state conditions are being approached.
SPVP 70	Also calculate the total current *effective* viscoplastic strain $\bar{\epsilon}_{vp}{}^{n+1}$ according to (8.47).
SPVP 71–76	Evaluate the current viscoplastic flow rate $\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1}$ according to (8.7).
SPVP 77	If the Gauss point is elastic, avoid calculation of the new time step length.

SPVP 78–79 Calculate $\bar{\dot{\epsilon}}_{vp}{}^{n+1}$, the effective value of the viscoplastic strain rate.

SPVP 80–81 Calculate $\bar{\epsilon}^{n+1}$, the total effective strain.

SPVP 82–83 Evaluate the interval length for the next time step according to (8.29) as

$$\Delta t_{n+1} = \tau\left[\frac{\bar{\epsilon}^{n+1}}{\bar{\dot{\epsilon}}_{vp}{}^{n+1}}\right]^{1/2}_{\min},$$

where TFACT is the parameter τ and the minimum value of Δt_{n+1} is taken with respect to all Gauss points throughout the structure.

SPVP 84–85 Termination of loops over Gauss points and elements respectively.

SPVP 87 For the first time step of a load increment reset the step length equal to the initial value input.

SPVP 88 Zero the Gauss point counter over all elements.

SPVP 89 Loop over each element.

SPVP 90 Identify the element material property number.

SPVP 91–96 Store the element coordinates in array ELCOD.

SPVP 97 Identify the element thickness.

SPVP 98 Zero the local Gauss point counter.

SPVP 99–102 Enter the loops for numerical integration and evaluate the local coordinates (ξ, η) at the sampling point.

SPVP 103–104 Increment the local and global Gauss point counters.

SPVP 105 Evaluate the shape functions and their local derivatives.

SPVP 106 Evaluate the Gauss point coordinates, determinant of the Jacobian matrix and the Cartesian derivatives of the shape functions.

SPVP 107–109 Calculate the elemental volume for numerical integration.

SPVP 110 Evaluate the $\boldsymbol{B}$ matrix.

SPVP 111 Evaluate the $\boldsymbol{D}$ matrix.

SPVP 112–113 Store the total current stresses σ^{n+1} locally in STRES.

SPVP 114–115 Calculate the deviatoric stresses and J_2'.

SPVP 116–117 For the implicit or semi-implicit time stepping scheme evaluate $\hat{\boldsymbol{D}}^{n+1}$.

SPVP 121–125 Calculate $\hat{\boldsymbol{D}}^{n+1}\dot{\epsilon}_{vp}{}^{n+1}\Delta t_{n+1}$ and store locally in STRES.

SPVP 126–132 Evaluate the pseudo loads to be applied for the next timestep, $\Delta \boldsymbol{V}^{n+1}$ according to (8.28) and (8.34) as

$$\Delta V^{n+1} = \int_{\Omega} B^T\{\hat{D}^{n+1}\dot{\epsilon}_{vp}^{n+1}\Delta t_{n+1} + \sigma^{n+1}\}d\Omega + f^{n+1}.$$

SPVP 133–134 Termination of loops over Gauss points and elements respectively.

SPVP 135–137 Complete the computations of SPVP 126–132 by adding the term f^{n+1}.

Subroutine INVAR which calculates the deviatoric stresses and J_2' is identical to that employed in Chapter 7 for elasto-plastic problems and is described in detail in Section 7.8.3. Subroutine YIELDF has been previously described in Section 7.8.4.1.

8.9 Subroutine FLOWVP

The function of this subroutine is to determine the viscoplastic strain rate according to (8.7).

Subroutine FLOWVP is now presented and described.

```
      SUBROUTINE FLOWVP(AVECT,PROPS,LPROP,STEFF,NSTR1,MTOTG,VIVEL,
     .                  YIELD,KGAUS,MMATS,NCRIT,FNORM,ALLOW)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE VISCOPLASTIC STRAIN RATE
C
C***********************************************************************
      DIMENSION AVECT(4),PROPS(MMATS,10),VIVEL(5,MTOTG)
      ALLOW=0.01
      IF(STEFF.EQ.0.0) GO TO 90
      YOUNG=PROPS(LPROP,1)
      POISS=PROPS(LPROP,2)
      HARDS=PROPS(LPROP,6)
      FRICT=PROPS(LPROP,7)
      GAMMA=PROPS(LPROP,8)
      DELTA=PROPS(LPROP,9)
      NFLOW=PROPS(LPROP,10)
      ROOT3=1.73205080757
      FDATM=PROPS(LPROP,5)
      FRICT=FRICT*0.017453292
      IF(NCRIT.EQ.3) FDATM=FDATM*COS(FRICT)
      IF(NCRIT.EQ.4) FDATM=6.0*FDATM*COS(FRICT)/(ROOT3*(3.0-SIN(FRICT)))
      IF(HARDS.GT.0.0) FDATM=FDATM+VIVEL(5,KGAUS)*HARDS
      IF(FDATM.LT.0.001) FDATM=1.0
      FCURR=YIELD-FDATM
      FNORM=FCURR/FDATM
      IF(FNORM.LT.ALLOW) GO TO 90
      IF(NFLOW.EQ.1) GO TO 50
      CMULT=GAMMA*(EXP(DELTA*FNORM)-1.0)
      GO TO 60
   50 CMULT=GAMMA*(FNORM**DELTA)
   60 DO 70 ISTR1=1,NSTR1
   70 AVECT(ISTR1)=CMULT*AVECT(ISTR1)
      DO 80 ISTR1=1,NSTR1
   80 VIVEL(ISTR1,KGAUS)=AVECT(ISTR1)
      RETURN
   90 DO 100 ISTR1=1,NSTR1
  100 VIVEL(ISTR1,KGAUS)=0.0
      RETURN
      END
```

FLVP 9 Specify ALLOW, the permitted tolerance by which the stress point is allowed to deviate from the yield surface.

FLVP 10 For the (unlikely) case of a Gauss point with zero stress (identified by $J_2' = J_3' = 0$) avoid all viscoplastic calculations.

FLVP 11 Identify YOUNG as the elastic modulus, E.

FLVP 12 Identify POISS as the Poissons ratio, ν.

FLVP 13 Identify HARDS as H' for linear strain hardening.

FLVP 14 Identify FRICT as the friction angle ϕ for Mohr–Coulomb and Drucker–Prager materials.

FLVP 15 Identify GAMMA as the fluidity parameter, γ.

FLVP 16 Identify DELTA as the index M in (8.8) or N in (8.9), according to the flow function specified.

FLVP 17 Identify NFLOW as the parameter specifying type of flow function:

NFLOW = 0 – flow function (8.8) to be used,
NFLOW = 1 – flow function (8.9) to be used.

FLVP 18 Compute $\sqrt{(3)}$.

FLVP 19–22 Identify FDATM as the effective yield stress, $\sigma_Y{}^0$, according to Column 4, Table 7.2.

FLVP 23 Evaluate the current yield stress as $F_0 = \sigma_Y{}^0 + H'\bar{\epsilon}_{vp}$, where $\bar{\epsilon}_{vp}$ is the current effective viscoplastic strain, according to (8.47).

FLVP 24 For elastic creep problems, solved by setting $F_0 = 0$, reset F_0 as a low value to avoid overflow in (8.8) and (8.9).

FLVP 25–26 Calculate $(F-F_0)/F_0$ where F is the effective stress value evaluated as YIELD in subroutine INVAR.

FLVP 27 If $(F-F_0)/F_0$ is less than ALLOW avoid any further viscoplastic calculations, i.e. the stress point is assumed to be sufficiently close to the yield surface.

FLVP 29 Evaluate $\gamma\langle\Phi\rangle$ for flow function (8.8).

FLVP 31 Evaluate $\gamma\langle\Phi\rangle$ for flow function (8.9).

FLVP 32–35 Use flow vector $\boldsymbol{a}$ to form $\dot{\boldsymbol{\epsilon}}_{vp}{}^{n+1} = \gamma\langle\Phi\rangle\boldsymbol{a}^{n+1}$.

FLVP 37–38 For elastic points only, set the viscoplastic strain rate to zero.

8.10 Subroutine STRESS

The function of this subroutine is to evaluate the increment in stress occurring during a time step according to (8.20).

Subroutine STRESS is presented below:

```
      SUBROUTINE STRESS(DMATX,LPROP,NTYPE,PROPS,NDOFN,CARTD,ELDIS,
     .                  SHAPE,GPCOD,NSTRE,VIVEL,DTIME,STRSG,KGASP,
     .                  MTOTG,MMATS,SVECT,NNODE,NSTR1,KGAUS,TLDIS)
C*********************************************************************
C
C**** EVALUATE THE INCREMENTS OF STRAIN AND STRESS
C
C*********************************************************************
```

```
      DIMENSION SVECT(4),PROPS(MMATS,10),ELDIS(2,9),CARTD(2,9),
     .          DMATX(4,4),AGASH(2,2),STRES(4),STRAN(4),STRSG(4,MTOTG),
     .          SHAPE(9),VIVEL(5,MTOTG),TLDIS(2,9),CGASH(2,2),
     .          GPCOD(2,9)
      POISS=PROPS(LPROP,2)
      DO 10 IDOFN=1,NDOFN
      DO 10 JDOFN=1,NDOFN
      BGASH=0.0
      DGASH=0.0
      DO 20 INODE=1,NNODE
      DGASH=DGASH+CARTD(JDOFN,INODE)*TLDIS(IDOFN,INODE)
   20 BGASH=BGASH+CARTD(JDOFN,INODE)*ELDIS(IDOFN,INODE)
      CGASH(IDOFN,JDOFN)=BGASH
   10 AGASH(IDOFN,JDOFN)=DGASH
C
C*** CALCULATE THE TOTAL AND INCREMENTAL STRAINS
C
      SVECT(1)=AGASH(1,1)
      SVECT(2)=AGASH(2,2)
      SVECT(3)=AGASH(1,2)+AGASH(2,1)
      IF(NTYPE.NE.3) GO TO 70
      SVECT(4)=0.0
      DO 60 INODE=1,NNODE
      SVECT(4)=SVECT(4)+TLDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP)
   60 CONTINUE
   70 CONTINUE
      STRAN(1)=CGASH(1,1)
      STRAN(2)=CGASH(2,2)
      STRAN(3)=CGASH(1,2)+CGASH(2,1)
      IF(NTYPE.NE.3) GO TO 90
      STRAN(4)=0.0
      DO 80 INODE=1,NNODE
      STRAN(4)=STRAN(4)+ELDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP)
   80 CONTINUE
   90 CONTINUE
      DO 50 ISTRE=1,NSTRE
   50 STRAN(ISTRE)=STRAN(ISTRE)-VIVEL(ISTRE,KGAUS)*DTIME
C
C*** AND THE INCREMENTAL STRESSES
C
      DO 30 ISTRE=1,NSTRE
      STRES(ISTRE)=0.0
      DO 30 JSTRE=1,NSTRE
   30 STRES(ISTRE)=STRES(ISTRE)+DMATX(ISTRE,JSTRE)*STRAN(JSTRE)
      IF(NTYPE.EQ.1) STRES(4)=0.0
      IF(NTYPE.EQ.2) STRES(4)=POISS*(STRES(1)+STRES(2))
      DO 40 ISTR1=1,NSTR1
   40 STRSG(ISTR1,KGAUS)=STRSG(ISTR1,KGAUS)+STRES(ISTR1)
      RETURN
      END
```

STRS 13 Identify POISS as the material Poisson's ratio.
STRS 14–22 Evaluate the Cartesian derivatives of both the displacement increment and the total displacement.
STRS 26–33 Evaluate the total and incremental strains $\boldsymbol{B}\boldsymbol{d}^n$ and $\boldsymbol{B}\Delta\boldsymbol{d}^n$.
STRS 34–45 Calculate the elastic portion of the strains, $\boldsymbol{B}\Delta\boldsymbol{d}^n - \dot{\boldsymbol{\epsilon}}_{vp}{}^n \Delta t_n$.
STRS 49–52 Calculate the stresses according to (8.20).
STRS 53–54 For plane stress and plane strain problems evaluate the out-of-plane stress component.
STRS 55–56 Finally calculate the total current stress as $\sigma^{n+1} = \sigma^n + \Delta\sigma^n$.

8.11 Subroutine ZERO

This subroutine performs the same task as the subroutine described in Section 7.8.2 for elasto-plastic problems. It merely initializes to zero some arrays required for the accumulation of data. Subroutine ZERO is presented below without further comment.

```
      SUBROUTINE ZERO(ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM,  ZRO2
     .                NEVAB,NGAUS,NSTR1,NTOTG,NTOTV,NVFIX,STRSG,        ZRO2
     .                TDISP,VIVEL,VISTN,TTIME,TLOAD,TREAC,              ZRO2
     .                TFACT,MVFIX)                                      ZRO2
C*********************************************************************** ZRO2
C                                                                       ZRO2
C**** THIS SUBROUTINE INITIALISES VARIOUS ARRAYS TO ZERO                ZRO2
C                                                                       ZRO2
C*********************************************************************** ZRO2
      DIMENSION ELOAD(MELEM,MEVAB),STRSG(4,MTOTG),TDISP(MTOTV),          ZRO2
     .          TLOAD(MELEM,MEVAB),TREAC(MVFIX,2),VIVEL(5,MTOTG),        ZRO2
     .          VISTN(4,MTOTG)                                          ZRO2
      TTIME=0.0                                                         ZRO2
      TFACT=0.0                                                         ZRO2
      DO 30 IELEM=1,NELEM                                               ZRO2
      DO 30 IEVAB=1,NEVAB                                               ZRO2
      ELOAD(IELEM,IEVAB)=0.0                                            ZRO2
   30 TLOAD(IELEM,IEVAB)=0.0                                            ZRO2
      DO 40 ITOTV=1,NTOTV                                               ZRO2
   40 TDISP(ITOTV)=0.0                                                  ZRO2
      DO 50 IVFIX=1,NVFIX                                               ZRO2
      DO 50 IDOFN=1,NDOFN                                               ZRO2
   50 TREAC(IVFIX,IDOFN)=0.0                                            ZRO2
      DO 60 ITOTG=1,NTOTG                                               ZRO2
      VIVEL(5,ITOTG)=0.0                                                ZRO2
      DO 60 ISTR1=1,NSTR1                                               ZRO2
      VISTN(ISTR1,ITOTG)=0.0                                            ZRO2
      VIVEL(ISTR1,ITOTG)=0.0                                            ZRO2
   60 STRSG(ISTR1,ITOTG)=0.0                                            ZRO2
      RETURN                                                            ZRO2
      END                                                               ZRO2
```

8.12 Subroutine STEADY for monitoring steady state convergence

The role of this subroutine is to check whether or not steady state conditions have been achieved at the end of each time step. Convergence to a steady state condition is monitored according to the increment in viscoplastic strain which occurs during the time step. For checking purposes the effective viscoplastic strain rate, $\bar{\dot{\epsilon}}_{vp}{}^{n+1}$, defined by (8.47) is employed and steady state conditions are deemed to have been achieved at the end of time step n, if

$$\left(\Delta t_{n+1} \sum_{\substack{\text{All Gauss}\\ \text{points}}} \bar{\dot{\epsilon}}_{vp}{}^{n+1} \Big/ \Delta t_1 \sum_{\substack{\text{All Gauss}\\ \text{points}}} \bar{\dot{\epsilon}}_{vp}{}^{1}\right) \times 100 \leqslant \text{TOLER}, \quad (8.48)$$

where TOLER is a convergence tolerance value prescribed as input in Subroutine INCREM, described in Section 6.5.3. From (8.48) it is seen that a *global* measure of convergence is taken in the subroutine presented in this section. A *local* steady state convergence condition could alternatively be enforced by requiring (8.48) to be satisfied for each Gauss point in the structure which is yielding viscoplastically.

The structure of this subroutine is identical to that of subroutine CONVP, presented in Section 4.9, for one-dimensional structures.

Subroutine STEADY is now presented.

```
      SUBROUTINE STEADY(NELEM,NGAUS,NCHEK,VIVEL,ISTEP,FIRST,TOLER,PVALU,
     .                  MTOTG,DTIME,NSTR1,TTIME)
C***********************************************************************
C
C**** THIS SUBROUTINE CHECKS FOR ATTAINMENT OF STEADY STATE CONDITIONS
C
C***********************************************************************
      DIMENSION VIVEL(5,MTOTG),DESTN(4)
      NCHEK=1
      NTOTG=NELEM*NGAUS*NGAUS
      TOTAL=0.0
      DO 10 ITOTG=1,NTOTG
      DO 40 ISTR1=1,NSTR1
   40 DESTN(ISTR1)=VIVEL(ISTR1,ITOTG)*DTIME
   10 TOTAL=TOTAL+SQRT((2.0*(DESTN(1)*DESTN(1)+DESTN(2)*DESTN(2)+
     . DESTN(4)*DESTN(4))+DESTN(3)*DESTN(3))/3.0)
      IF(ISTEP.EQ.1) FIRST=TOTAL
      IF(FIRST.EQ.0.0) GO TO 15
      RATIO=100.0*TOTAL/FIRST
      GO TO 25
   15 RATIO=0.0
   25 CONTINUE
      IF(ISTEP.EQ.1) GO TO 20
      IF(RATIO.LE.TOLER) NCHEK=0
      IF(RATIO.GT.PVALU) NCHEK=999
   20 PVALU=RATIO
      WRITE(6,900) TTIME
  900 FORMAT(1H0,5X,12HTOTAL TIME =,E17.6)
      WRITE(6,30) NCHEK,RATIO,REMAX
   30 FORMAT(1H0,3X,18HCONVERGENCE CODE =,I4,3X,28HNORM OF RESIDUAL SUM
     .RATIO =,E14.6,3X,18HMAXIMUM RESIDUAL =,E14.6)
      RETURN
      END
```

8.13 The main, master or controlling segment

This segment controls the timestepping process and accesses all the other subroutines appropriately. In particular it controls the incremention of the applied loads and the output of results at selected time intervals. The frequency of output is controlled by means of two parameters NOUTP(1) and NOUTP(2) which are specified as input data for every load increment in subroutine INCREM described in Section 6.5.3. The precise specification of these parameters is however somewhat different for the present application. In this case NOUTP(1) controls the frequency of output of the displacements and NOUTP(2) the frequency of output of the stresses and viscoplastic strains. In particular, if NOUTP(1) is specified as 7 for a particular load increment, then the displacements will be output every 7th timestep within that increment. This is accomplished by evaluating for every timestep, ISTEP, the quantity

(ISTEP/NOUTP(1))*NOUTP(1)

and then checking this value against ISTEP. The two will be equal only when ISTEP is an exact multiple of NOUTP(1). A similar check for stress output is undertaken for NOUTP(2).

The parameter MSTEP specifies the maximum number of timesteps to be considered for the load increment. If steady state conditions are achieved before MSTEP timesteps, the next load increment, is applied immediately condition (8.48) is satisfied.

The role of the load incrementing factor, FACTO, is identical to that described in Section 6.5.3.

In this segment input data is also received which controls the timestepping algorithm to be employed. The following information is input:

TIMEX Parameter, Θ, which controls the type of timestepping algorithm to be employed:
TIMEX = 0.0—Explicit scheme,
= 0.5—Semi-implicit or trapezoidal scheme,
= 1.0—Fully implicit.

TAUFT The parameter τ discussed in Section 8.3.

DTINT The initial time step length. This specifies the step length for the first time step of each load increment. The time step length needs to be readjusted at the beginning of a new load increment since the step length computed as steady state conditions are approached in the previous time step will in general be too large.

FTIME The factor by which it is attempted to increase the step length from the value used for the previous time step. This parameter is generally input as 1.5 as mentioned in Section 8.3.

The following channel numbers are employed by the program: 5 (card reader), 6 (line printer), 1, 2, 3, 4, 8 (scratch files). This main segment is now presented and descriptive notes provided where necessary.

```
      MASTER VISCO
C*********************************************************************
C     PROGRAM FOR THE ELASTO-VISCOPLASTIC ANALYSIS OF PLANE STRESS,
C     PLANE STRAIN AND AXISYMMETRIC SOLIDS
C*********************************************************************
      DIMENSION ASDIS(120),COORD(60,2),ELOAD(20,18),ESTIF(18,18),
     .          EQRHS(10),EQUAT(40,10),FIXED(120),
     .          GLOAD(40),GSTIF(986),
     .          IFFIX(120),LNODS(20,9),LOCEL(18),MATNO(20),
     .          NACVA(40),NAMEV(10),NDEST(18),NDFRO(20),NOFIX(25),
     .          NOUTP(2),NPIVO(10),
     .          POSGP(4),PRESC(25,2),PROPS(5,10),RLOAD(20,18),
     .          STFOR(120),TREAC(25,2),VECRV(40),WEIGP(4),
     .          STRSG(4,180),TDISP(120),
     .          TLOAD(20,18),VIVEL(5,180),VISTN(4,180)
C
C*** PRESET VARIABLES ASSOCIATED WITH DYNAMIC DIMENSIONING
C
      CALL DIMEN(MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,MSTIF,MTOTG,MTOTV,
     .           MVFIX,NDOFN,NPROP,NSTRE)
C
C*** CALL THE SUBROUTINE WHICH READS MOST OF THE PROBLEM DATA
C
      CALL INPUT(COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,MMATS,
     .           MPOIN,MTOTV,MVFIX,NALGO,
     .           NCRIT,NDFRO,NDOFN,NELEM,NEVAB,NGAUS,NGAU2,
     .           NINCS,NMATS,NNODE,NOFIX,NPOIN,NPROP,NSTRE,
     .           NSTR1,NTOTG,NTOTV,
     .           NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP)
C
C*** CALL THE SUBROUTINE WHICH COMPUTES THE CONSISTENT LOAD VECTORS
C    FOR EACH ELEMENT AFTER READING THE RELEVANT INPUT DATA
C
      CALL LOADPS(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
     .            NEVAB,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
     .            PROPS,RLOAD,WEIGP,NDOFN)
C
C*** INITIALISE CERTAIN ARRAYS
C
      CALL ZERO(ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM,
     .          NEVAB,NGAUS,NSTR1,NTOTG,NTOTV,NVFIX,STRSG,TDISP,
     .          VIVEL,VISTN,TTIME,TLOAD,TREAC,TFACT,MVFIX)
      READ(5,900)  TIMEX,TAUFT,DTINT,FTIME
      WRITE(6,910) TIMEX,TAUFT,DTINT,FTIME
  900 FORMAT(4F10.3)
  910 FORMAT(1H0,5X,25HTIME STEPPING PARAMETER =,F10.3,5X,
     . 28HTIME STEP STABILITY FACTOR =,F10.5,//
     . 5X,26HINITIAL TIME STEP LENGTH =,F10.5,5X,32HTIME STEP INCREMENT
     .PARAMETER = ,F10.5)
C
C*** LOOP OVER EACH INCREMENT
C
      DO 100 IINCS = 1,NINCS
C
C*** READ DATA FOR CURRENT INCREMENT
C
      CALL INCREM(ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,MTOTV,
     .            MVFIX,NDOFN,NELEM,NEVAB,NOUTP,NOFIX,NTOTV,
     .            NVFIX,PRESC,RLOAD,TFACT,TLOAD,TOLER)
C
C*** LOOP OVER EACH ITERATION
C
      DTIME=0.0
      DO 50 ISTEP=1,MITER
```

```
      TTIME=TTIME+DTIME
C
C*** CALL ROUTINE WHICH SELECTS SOLUTION ALORITHM VARIABLE KRESL
C
      CALL ALGOR(FIXED,IINCS,ISTEP,KRESL,TIMEX,MTOTV,NALGO,NTOTV)
C*** CHECK WHETHER A NEW EVALUATION OF THE STIFFNESS MATRIX IS REQUIRED
C
      IF(KRESL.EQ.1) CALL STIFVP(COORD,IINCS,LNODS,MATNO,
     .          MEVAB,MMATS,MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE,
     .         NSTRE,NSTR1,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     .         STRSG,NTYPE,NCRIT,TIMEX,DTIME)

C
C*** SOLVE EQUATIONS
C
      CALL FRONT(ASDIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,ISTEP,
     .          GLOAD,GSTIF,LOCEL,LNODS,KRESL,MBUFA,MELEM,MEVAB,MFRON,
     .          MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,NDOFN,NELEM,NEVAB,
     .          NNODE,NOFIX,NPIVO,NPOIN,NTOTV,TDISP,TLOAD,TREAC,
     .          VECRV)
C
C*** CALCULATE RESIDUAL FORCES
C
      CALL STEPVP(ASDIS,COORD,ELOAD,ISTEP,LNODS,LPROP,TIMEX,
     .           MATNO,MELEM,MMATS,MPOIN,MTOTG,TAUFT,DTIME,
     .           MTOTV,NDOFN,NELEM,NEVAB,NGAUS,NNODE,NSTR1,
     .           NTYPE,POSGP,PROPS,NSTRE,NCRIT,STRSG,WEIGP,
     .           TDISP,VISTN,VIVEL,TLOAD,FTIME,DTINT,IINCS)
C
C*** CHECK FOR CONVERGENCE TO STEADY STATE
C
      CALL STEADY(NELEM,NGAUS,NCHEK,VIVEL,ISTEP,FIRST,TOLER,PVALU,
     .           MTOTG,DTIME,NSTR1,TTIME)
C
C*** OUTPUT RESULTS IF REQUIRED
C
      IF(NOUTP(1).EQ.0) GO TO 110
      KOUTD=(ISTEP/NOUTP(1))*NOUTP(1)
      KOUTS=(ISTEP/NOUTP(2))*NOUTP(2)
      IF(KOUTD.NE.ISTEP.OR.KOUTS.NE.ISTEP) GO TO 110
      KOUTP=2
      IF(KOUTS.EQ.ISTEP) KOUTP=3
      CALL OUTPUT(ISTEP,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .           NPOIN,NVFIX,STRSG,TDISP,TREAC,NTYPE,NCHEK,VIVEL,
     .           KOUTP)
  110 CONTINUE
C
C*** IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS
C
      IF(NCHEK.EQ.0) GO TO 75
   50 CONTINUE
C
C***
C
   75 CALL OUTPUT(ISTEP,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .           NPOIN,NVFIX,STRSG,TDISP,TREAC,NTYPE,NCHEK,VIVEL,
     .           KOUTP)
  100 CONTINUE
      STOP
      END
```

VISC 64	For each load increment, initialise the time step length.
VISC 65	Enter the time-stepping loop for the current load increment.
VISC 66	Compute the total time elapsed.
VISC 70	For the first timestep of the first load increment prepare for a full equation solution rather than a resolution for an explicit formulation. For the implicit or semi-implicit algorithm a complete equation solution is required each and every time-step.
VISC 73–85	Formulate the element stiffnesses and solve the resulting equations.
VISC 89–94	Calculate quantities at the end of the timestep and evaluate the loads for the next timestep.
VISC 98–99	Check for convergence of the time stepping process to steady state conditions.
VISC 103–105	Check to see if either displacement or stress output is required for this timestep.
VISC 106–107	Set KOUTP = 2 for displacement output only and KOUTP = 3 for both stress and displacement output.
VISC 108–110	Output the results.
VISC 115	If steady state conditions have been reached, output the converged results, increment the loads and proceed with the time-stepping process.

8.14 General comparison of implicit and explicit time integration schemes

Before discussing the general case of a two-dimensional continuum it is instructive to consider the behaviour of a single degree of freedom system. In particular we will consider the response of a simple linear Maxwell model, as illustrated in Fig. 8.2. This situation is equivalent to the uniaxial visco-plastic model when the initial yield or threshold value, F_0, is reduced to zero. Figure 8.2 shows the stress relaxation histories for different time integration schemes when the model is subjected to a constant total strain. It is observed that all results obtained using the fully implicit scheme ($\Theta = 1$) lie to one side of the theoretical solution while the semi-implicit method ($\Theta = \frac{1}{2}$) gives results which lie to either side of the true curve. It is also evident that the explicit method ($\Theta = 0$) gives an oscillatory solution with the rate of convergence decreasing as the time step stability limit is approached. However, in each case the steady state solution is eventually correctly predicted. For the solution of elasto-plastic problems by use of the viscoplastic algorithm it is only the steady state solution that is of importance. Similarly in problems of creep, the transient stage may not be of interest in itself, as long as the steady state values are correctly arrived at.

For problems which are geometrically linear the solution process simplifies considerably. The strain matrix $\boldsymbol{B}^n$ is then constant throughout the analysis and from (8.19) it is seen to be equal to $\boldsymbol{B}_0$. For solution by the explicit time

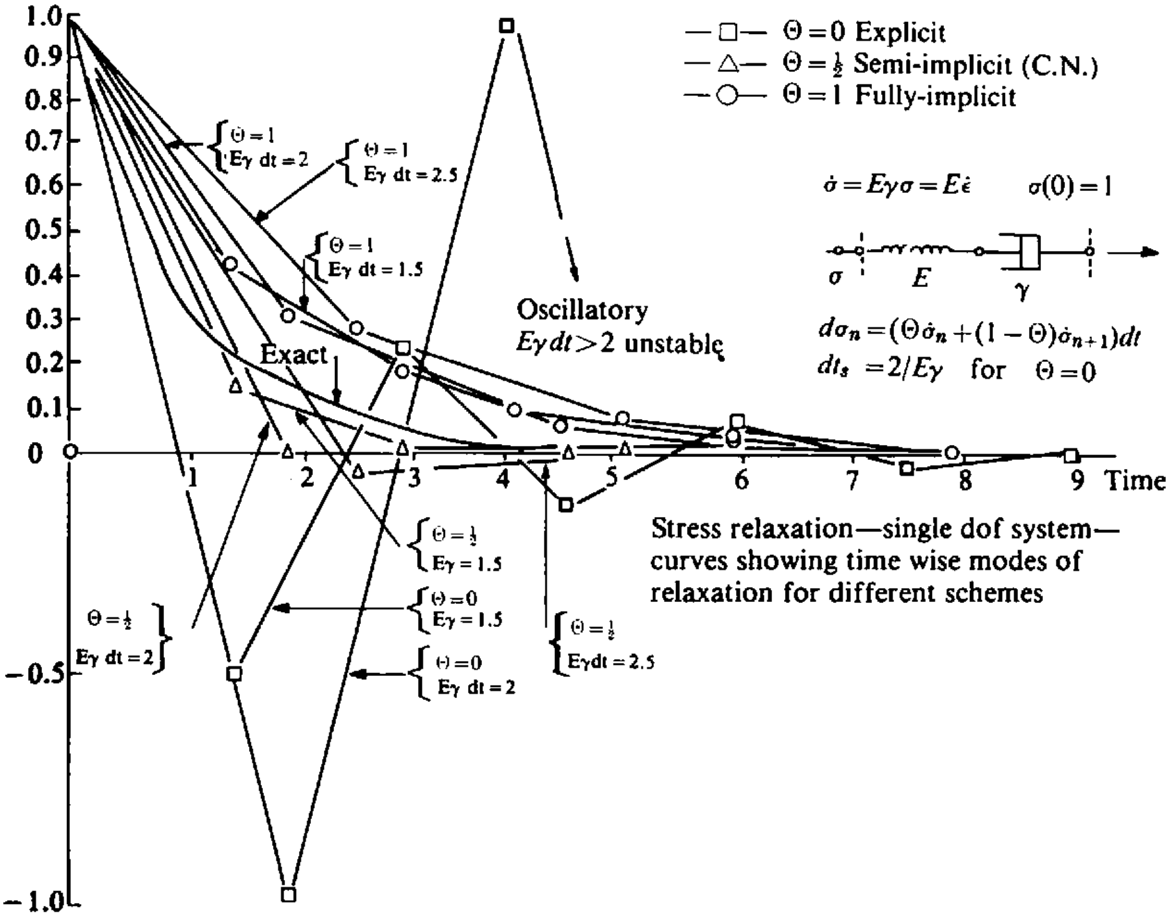


Fig. 8.2 Characteristics of explicit and implicit time stepping algorithms when applied to a linear Maxwell model.

marching scheme, $\Theta = 0$ and from (8.14) we have that $\boldsymbol{C}^n = \boldsymbol{0}$. Consequently, from (8.18), $\hat{\boldsymbol{D}}^n = \boldsymbol{D}$ and (8.24) implies that the tangential stiffness matrix becomes the linear elastic stiffness matrix and is constant throughout the solution process. Thus for the equation solution demanded by (8.23), a complete reduction and back-substitution is only required for the first time step and subsequent time intervals only require equation resolution.

Experience to date[2] indicates that solution by the implicit method increases the computation time by approximately a factor of 4–5 in comparison with the explicit approach, for the same solution tolerance factor (or time step length). This cost differential must be balanced against the greater time step lengths permitted by the unconditionally stable implicit method. However, increasing the time step length beyond prescribed limits results in a deterioration in solution accuracy. Where a variable stiffness approach is employed for some other reasons, such as to include geometric nonlinearity effects or time dependent material properties, solution by an implicit scheme entails little or no additional computing effort and such an approach is particularly advantageous. Modification of the program presented to account for large deformation effects is set as an exercise to the reader in Section 8.17.

Implicit and explicit time integration schemes are considered further in Chapters 10 and 11 for the solution of dynamic transient problems.

8.15 The overlay method for improved material response

The viscoplastic model described in the previous sections gives a material response whose general form is in keeping with experimental observations. However the precise strain/time histories (or creep curves) of many real materials cannot be accurately represented by a simple viscoplastic model. This is particularly so for materials whose strain response curves are non-linear with regard to the applied stress level, so that a doubling of the applied stress does not result in twice the strain at any given time.

A more elaborate material response can be modelled by use of the so-called *overlay or mechanical sublayer method*[10-13] in which the solid to be analysed is assumed to be composed of several layers or overlays each of which undergoes the same deformation. The total stress field is obtained by summing the different contributions of each overlay. By introducing a suitable number of overlays and assigning different material characteristics to each, a variety of sophisticated composite actions can be reproduced. In this section it is demonstrated how time-dependent overlay models can be used to simulate some experimentally observed material behaviours.

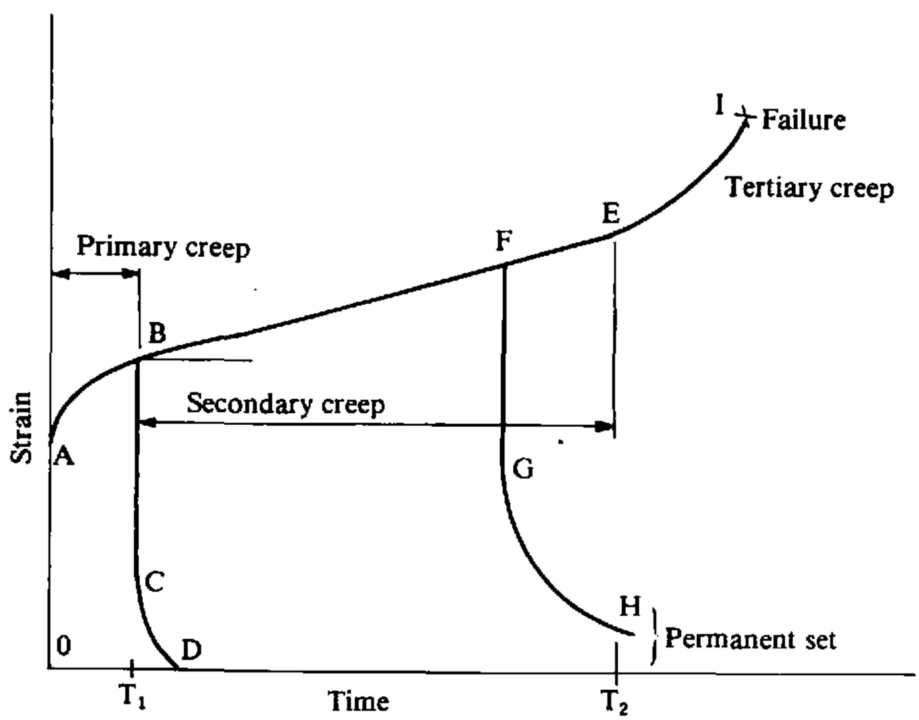


Fig. 8.3 Strain/time relationship at constant stress for many typical materials.

The strain-time relationship at constant stress which most materials exhibit to some degree or other is illustrated in Fig. 8.3. The instantaneous elastic strain, OA, is followed by a primary creep AB during which if unloading takes place an instantaneous elastic recovery results, followed by delayed elastic recovery, CD. If the load is not removed at time T_1 secondary creep begins which is accompanied by permanent deformation. Unloading at any time on the curve BE leaves a permanent set in the material. On continued loading past time T_2 tertiary creep begins, leading almost inevitably to failure.

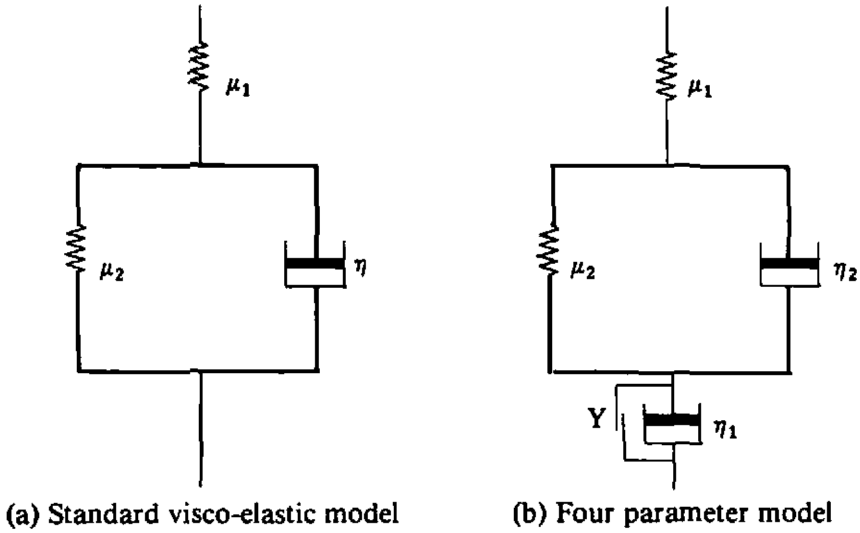


Fig. 8.4 Material models for simulation of the material behaviour of Fig. 8.3. (a) Standard visco-elastic model. (b) Four parameter model.

This behaviour can be numerically simulated by use of the rheological models shown in Fig. 8.4. The standard linear solid illustrated in Fig. 8.4(a) provides a visco-elastic response and represents the behaviour of the material up to time T_1. After this time the behaviour is closely approximated by the five parameter model shown in Fig. 8.4(b) where a friction slider component in parallel with a viscous dashpot has been added. This component becomes active only if the applied stress exceeds some limiting value, Y and the friction slider provides the permanent deformation or viscoplastic effect. For use in the overlay method it is desirable to consider 'Maxwell equivalents' of these models. Figure 8.5(a) shows the equivalent model to that of Fig. 8.4(a) both being governed by the differential equation

$$p_1 D\sigma + p_0 \sigma = q_1 D\epsilon + q_0 \epsilon, \tag{8.49}$$

where p_i and q_i are constants and D denotes the differential operator with respect to time. Similarly Fig. 8.5(b) illustrates the Maxwell equivalent of Fig. 8.4(b), the governing equation for this case being

$$p_2 D^2\sigma + p_1 D\sigma + p_0 \sigma = q_2 D^2\epsilon + q_1 D\epsilon + q_0 \epsilon. \tag{8.50}$$

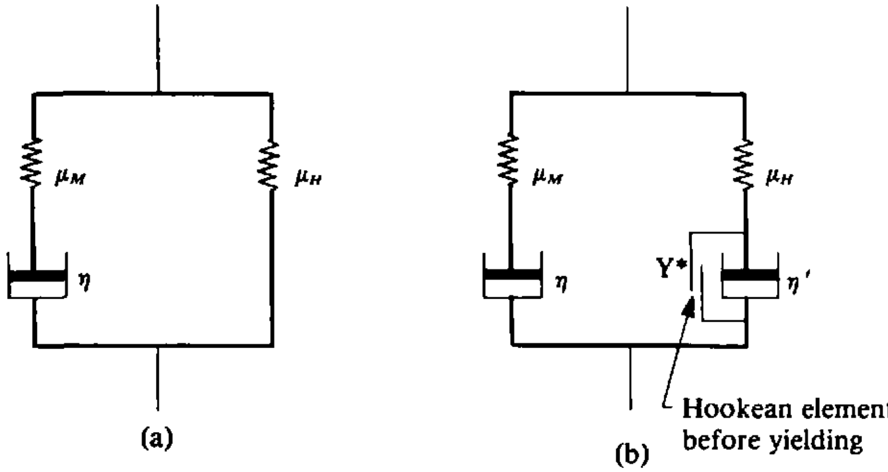


Fig. 8.5 Equivalent representation of the models of Fig. 8.4 using Maxwell type components.

The constants for the various components of the models in Figs. 8.4 and 8.5 are different but unique relationships exist. The configurations of Fig. 8.5 immediately suggest the use of overlay models. By employing at least one viscoplastic overlay and one Maxwell overlay (i.e. setting the threshold uniaxial yield value, $F_0 = 0$) the complete behaviour in the visco-elastic range as well as irrecoverable creep deformation can be generated. The model behaves as a 'standard linear solid' until failure of the friction slider in the visco-plastic overlay after which it behaves as a four parameter solid. In fact a fifth parameter, the yield limit of the slider must also be defined. These parameters are material characteristics and their values must be experimentally determined.

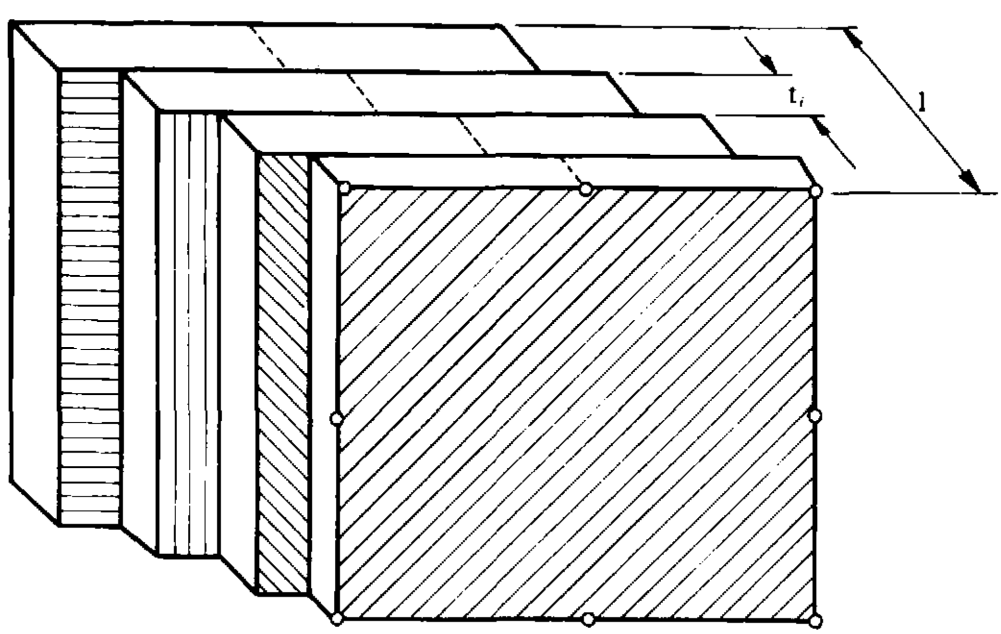


Fig. 8.6 The overlay model in two-dimensional situations.

8.15.1 Basic expressions of the overlay concept

The overlay model in a two-dimensional situation is illustrated schematically in Fig. 8.6. Each overlay can have a different thickness and material behaviour. With the nodes in each overlay coincidental, the same strain pattern is produced in each component. This results in a different stress field $\boldsymbol{\sigma}_j$ in each layer which contribute to the total stress field $\boldsymbol{\sigma}$ according to the overlay thickness, t_j, so that

$$\boldsymbol{\sigma} = \sum_{j=1}^{k} \boldsymbol{\sigma}_j t_j, \tag{8.51}$$

in which k is the total number of overlays in the model, and

$$\sum_{j=1}^{k} t_j = 1. \tag{8.52}$$

The equilibrium equations (8.21) which must be satisfied at each stage become

$$\int_{\Omega} [\boldsymbol{B}^n]^T \sum_{j=1}^{k} \boldsymbol{\sigma}_j{}^n t_j \, d\Omega + \boldsymbol{f}^n = \boldsymbol{0}. \tag{8.53}$$

Also the element stiffnesses (8.24) are the sum of each overlay contribution so that

$$\boldsymbol{K}_T{}^n = \sum_{j=1}^{k} \int_{\Omega} [\boldsymbol{B}^n]^T (\boldsymbol{D}^n)_j \boldsymbol{B}^n \, d\Omega, \tag{8.54}$$

where $(\hat{\boldsymbol{D}}^n)_j$ is the value of $\hat{\boldsymbol{D}}^n$ for each overlay in turn. Matrix $(\hat{\boldsymbol{D}}^n)_j$ will differ from overlay to overlay according to the material properties of each. The solution process is then identical to that described in the preceding sections with stress and strain terms being calculated for each overlay separately. It should be noted that the viscoplastic strain in each overlay will generally be different due to differences in threshold yield values and flow rates but the total strains must be the same.

Although the name overlay model arises from the physical interpretation of the two-dimensional situation the technique is essentially a mathematical convenience and can be readily extended to three-dimensional problems. In such cases the thickness can no longer be interpreted as a physical quantity and becomes merely a weighting parameter for combining the contribution of individual overlays. Indeed this is also the case in two-dimensional problems where negative thicknesses can be employed to simulate strain-softening conditions.[12]

8.15.2 Overlay models for some standard material behaviours

In this section we reproduce some standard material responses by combining different viscoplastic components through the overlay concept.[13]

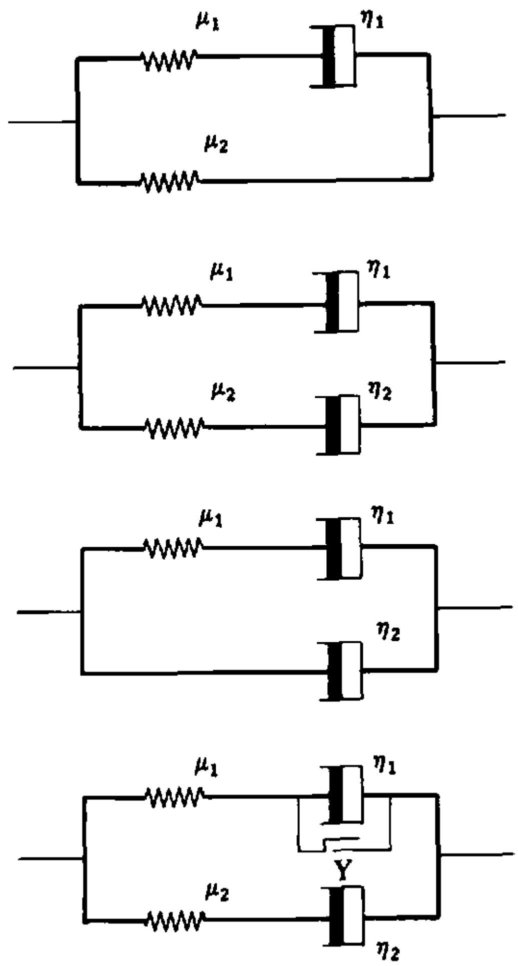


Fig. 8.7 Use of the overlay concept for the simulation of some standard material behaviours.

(i) *Visco-elastic response*

A two overlay model with F_0 set to zero for one overlay and infinitely large in the other reproduces a standard linear visco-elastic solid (Fig. 8.7). Any higher order time dependent constitutive relation can be simulated by the introduction of more overlays of the Maxwell type (i.e. $F_0 = 0$). Quite generally a stress–strain relationship of the form

$$\sum_{k=0}^{n} a_k D^k \sigma = \sum_{k=0}^{n} a_k D^k \epsilon, \tag{8.55}$$

in which a_k and b_k are real valued functions of the spatial coordinates and D denotes the differential time operator, can be modelled by the use of n Maxwell type overlays. The overlay approach reduces the n^{th} order differential equation (8.55) to n first order equations.

(ii) *Four parameter viscous model*

Two overlays with F_0 set to zero in each case provides a four parameter viscous model of the first kind (Fig. 8.7). Three overlays with F_0 set to (a) zero for one overlay (b) infinitely large for the second unit, (c) zero for the third overlay together with a small prescribed elastic modulus, reproduces a four parameter model of the second kind.

(iii) *Three element viscous model*

A two overlay model with F_0 set to zero in both and the elastic modulus assigned to be infinitely large in one reproduces the three element viscous model.

(iv) *Visco-elastic-plastic four parameter model*

This two overlay model is capable of reproducing the behaviour of most real engineering materials and is achieved by setting the threshold yield value of one overlay to zero. Before yielding of the friction slider, the material behaviour is visco-elastic followed by a viscoplastic response after initial yielding. By choosing the viscosity coefficients of the two dashpots appropriately the rate of straining after first yield can be controlled.

In order to illustrate how the combination of two simple material responses by the overlay method can simulate a more complex material behaviour the load cycling problem indicated in Fig. 8.8 is presented. One elastic (yield value set very large) and one viscoplastic overlay are considered. A static analysis of the load cycling of this model was performed by allowing steady state conditions to be achieved after application of each increment of load. The results are shown in Fig. 8.8 where the material properties employed are also included. A Bauschinger effect is immediately apparent on reversal of loading with yielding in compression commencing at a reduced value compared with initial yield in tension. Thus although each overlay has been assumed to be non-strain hardening with equal yield stress in tension and compression, the composite model exhibits a kinematic hardening behaviour.

As a further demonstration of the overlay approach, Fig. 8.9 shows how two overlays can be used to simulate the response of a real engineering material. The solid lines represent experimentally obtained creep curves for a rock salt and it is evident that the material behaviour is highly nonlinear with regard to the strain obtained at any time for a given applied load. The broken lines are the numerical material response obtained by using two overlays with material properties as shown in Fig. 8.9. The agreement obtained is acceptable for engineering purposes but a closer correspondence could be readily achieved by the use of additional overlays.

The main advantage of the overlay technique is that it allows the description of complex material behaviours by the use of components which individually exhibit a simple response.

All the program changes required to implement the overlay method in the viscoplastic program described earlier in this chapter are of a minor nature. Almost all the changes are associated with the summation process over each overlay demanded by (8.51), (8.53) and (8.54). Several array sizes must also be extended to allow separate storage of quantities for each overlay. Modification of the program is set as an exercise for the reader in Section 8.17.

8.16 Numerical examples

The first problem considered is the elasto-viscoplastic deformation of a thick tube under the action of internal pressure loading with the exterior surface remaining free. The mesh of Fig. 7.12(a) is employed in analysis with

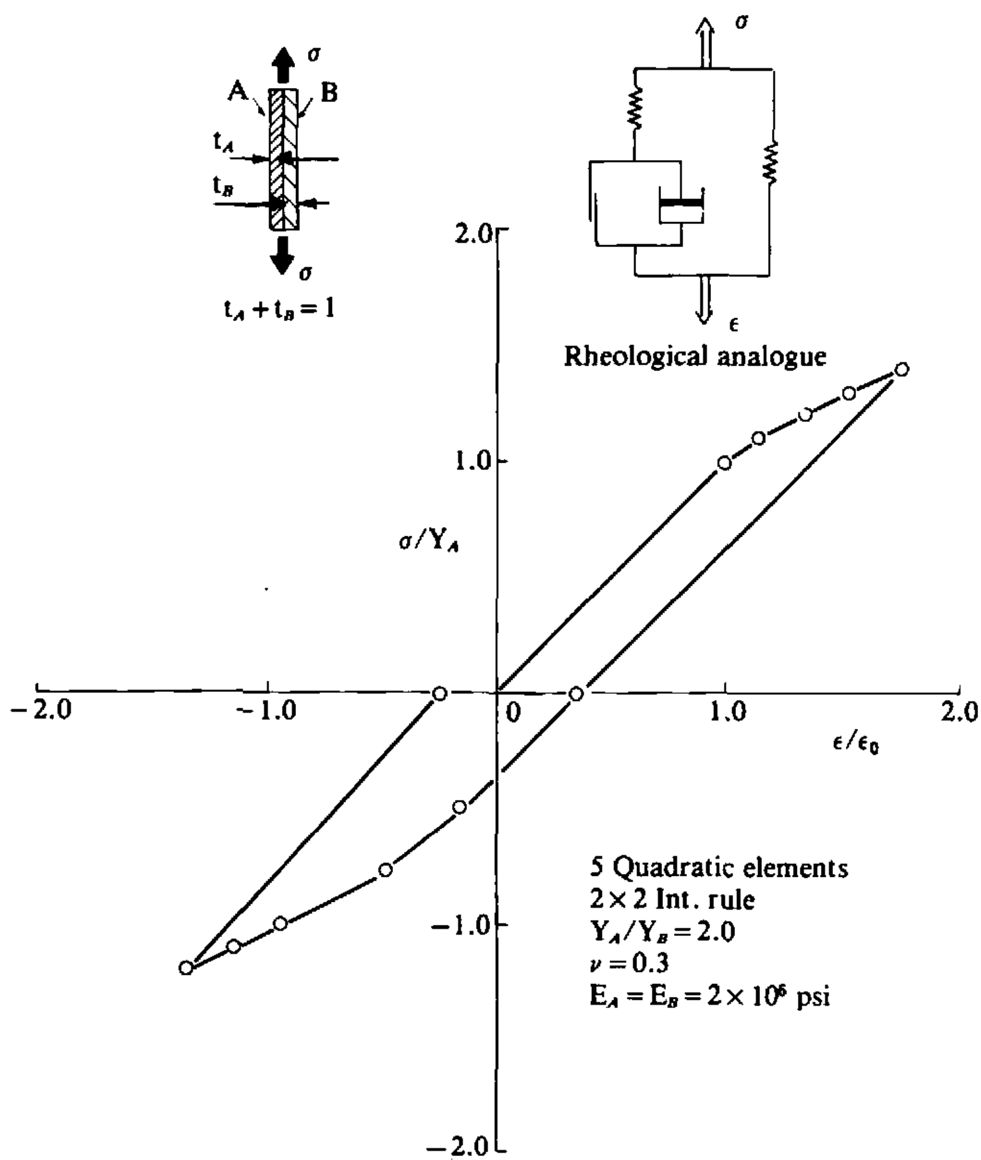


Fig. 8.8 Load cycling response of an overlay composite illustrating the Bauschinger effect.

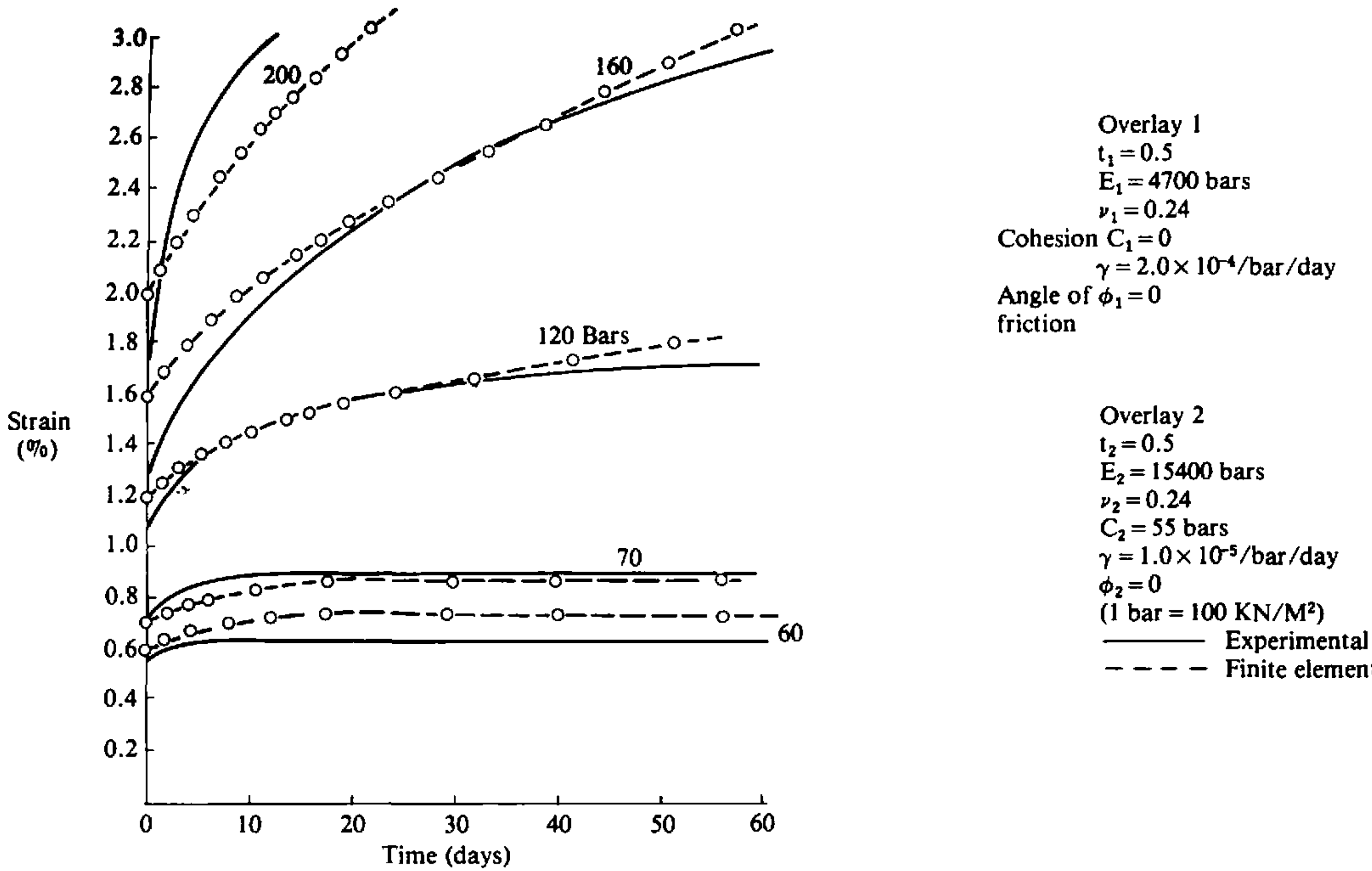


Fig. 8.9 Numerical simulation of experimental creep curves by use of the overlay method.

plane strain conditions being assumed in the axial direction. The material properties employed are identical to the case of Fig. 7.12(a) and the fluidity parameter is chosen as $\gamma = 0.001$. Again a Von Mises yield surface is adopted in solution and the flow function $\Phi(F) = F$ is assumed. An explicit time stepping algorithm ($\Theta = 0$) is initially employed and the radial displacement of the inner surface with time is shown in Fig. 8.10 for two increments of applied pressure. Steady state conditions are allowed to develop for an applied pressure of 12 dN/mm^2 before a further pressure increment of 2 dN/mm^2 is added. For each increment the time stepping parameter values $\tau = 0.01$, $k = 1.5$ were employed, the initial time-step length was chosen as 0.1 days and the steady state convergence tolerance parameter taken as 0.1%. Also shown in Fig. 8.10 are the results for the situation when an internal pressure of $P = 14$ dN/mm^2 is instantaneously applied. The steady state displacement is seen to be in good agreement with that obtained from the two-load

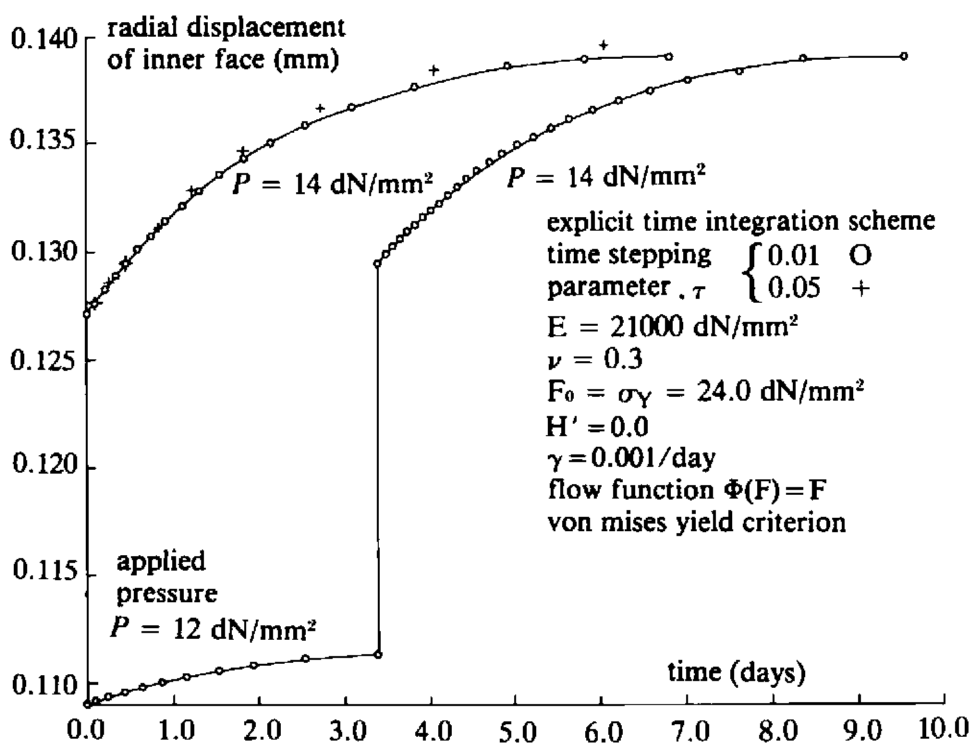


Fig. 8.10 Displacement of the inner surface with time of an elasto-viscoplastic cylinder subjected to an incrementally applied internal pressure (Mesh of Fig. 7.12(a)).

increment solution. The problem was reanalysed for an applied pressure, $P = 14$ dN/mm^2 using larger time-step lengths as governed by $\tau = 0.05$. The loss of accuracy is immediately apparent, with the larger time steps overestimating the viscoplastic strain rates.

The problem was then resolved using in turn, the implicit trapezoidal time stepping scheme ($\Theta = \frac{1}{2}$) and the full implicit or backward difference scheme ($\Theta = 1$). Good agreement between the three time integration schemes

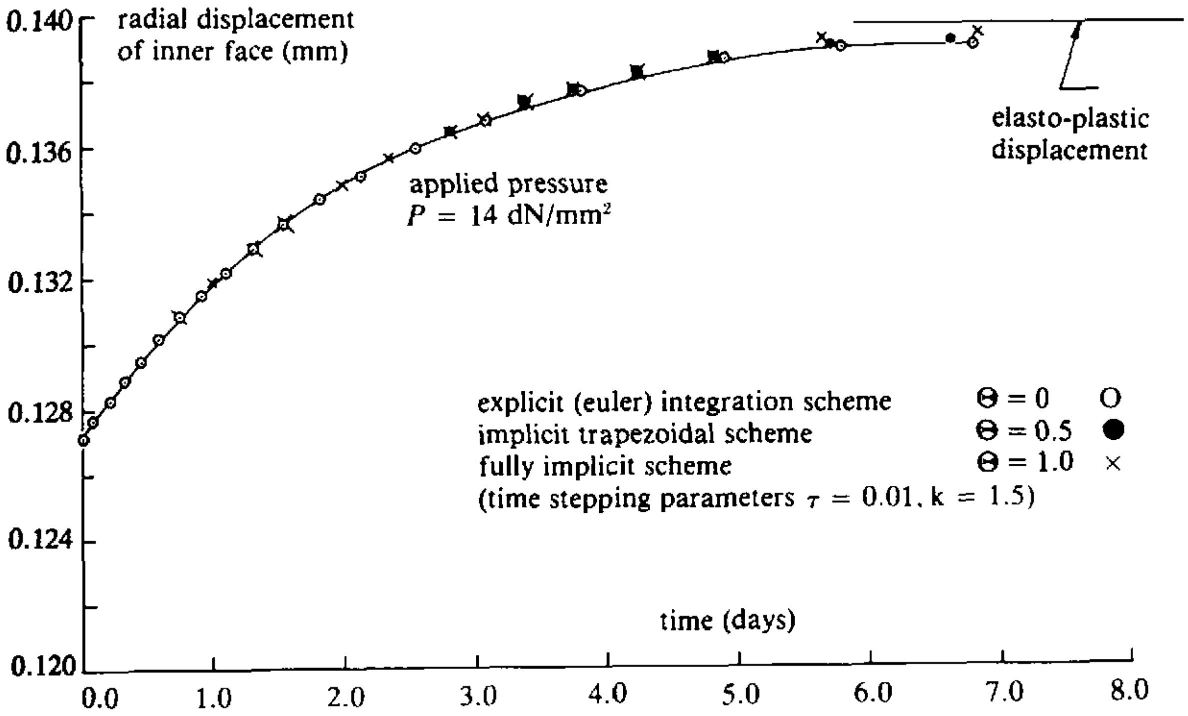


Fig. 8.11 Comparison of various time integration schemes for the internally pressurised cylinder of Fig. 8.10.

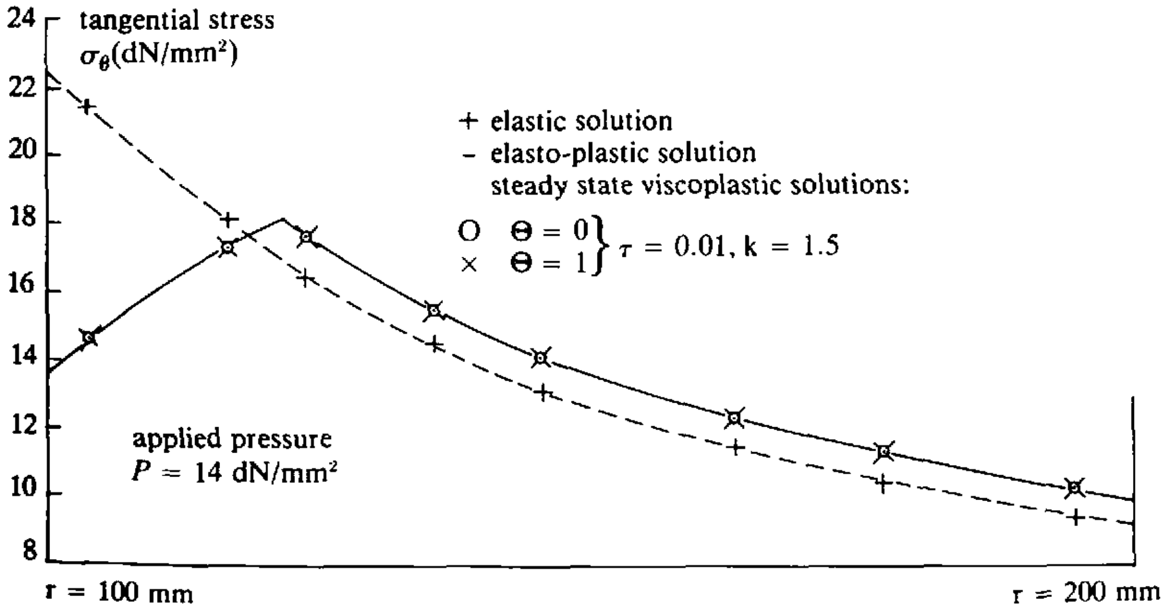


Fig. 8.12 Steady state tangential stress distribution in an elasto-viscoplastic internally pressurised cylinder.

is evident in Fig. 8.11 with the steady state displacement in each case comparing well with the corresponding elasto-plastic value of Fig. 7.12(b).

The steady state hoop stress distributions are shown in Fig. 8.12 for the time integration schemes $\Theta = 0$ and $\Theta = 1$, and the results are compared with the elasto-plastic solution of Fig. 7.13. Excellent agreement is obtained as required; since theoretically the steady state viscoplastic solution coincides with the corresponding elasto-plastic solution.

The problem of the stresses induced in the vicinity of an excavated underground storage cavity is illustrated in Fig. 8.13. Applications in this area include oil and gas reservoirs, nuclear waste disposal and geothermal energy problems. The cavity is assumed to be axisymmetric and Fig. 8.13

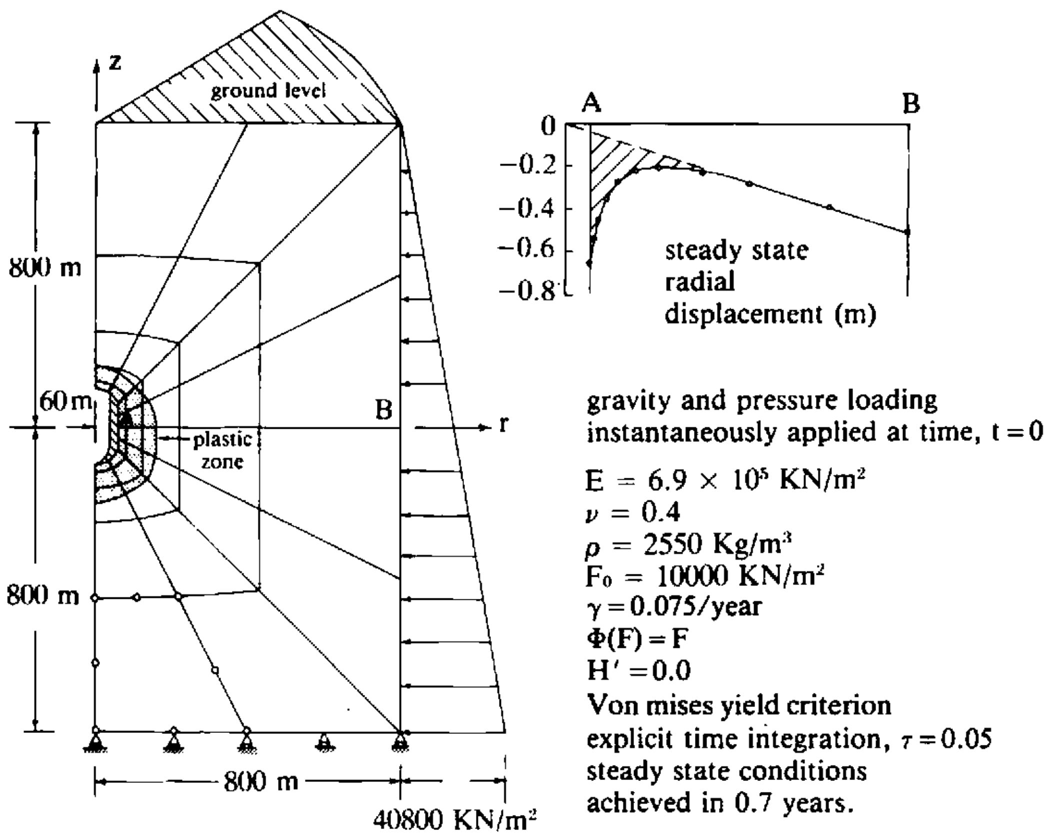


Fig. 8.13 Elasto-viscoplastic analysis of a subterranean cavity, showing zones of plasticity and steady state radial displacement at mid-height.

shows the finite element idealisation of a cylindrical portion of the surrounding rock mass. Before excavation of the cavity the tectonic stress field in the rock is assumed to be hydrostatic. This condition is simulated by a gravity loading together with a lateral hydrostatic pressure applied to the cylindrical face of the model. The material properties employed are indicated in Fig. 8.13. The cavity is assumed to be instantaneously excavated at time $t = 0$ and viscoplastic solution to steady state conditions performed by explicit time integration ($\Theta = 0$). Steady state conditions are achieved in 0.7 years and the zones of viscoplastic deformation at this time are illustrated in Fig. 8.13. It should be emphasised that since the fluidity parameter γ only enters the viscoplastic expressions through the product $\gamma . t$, then solution for different material fluidity values simply necessitates an adjustment of the time scale. Figure 8.13 also shows the radial displacement along section AB at steady state. The displacement distribution is seen to be made up of a

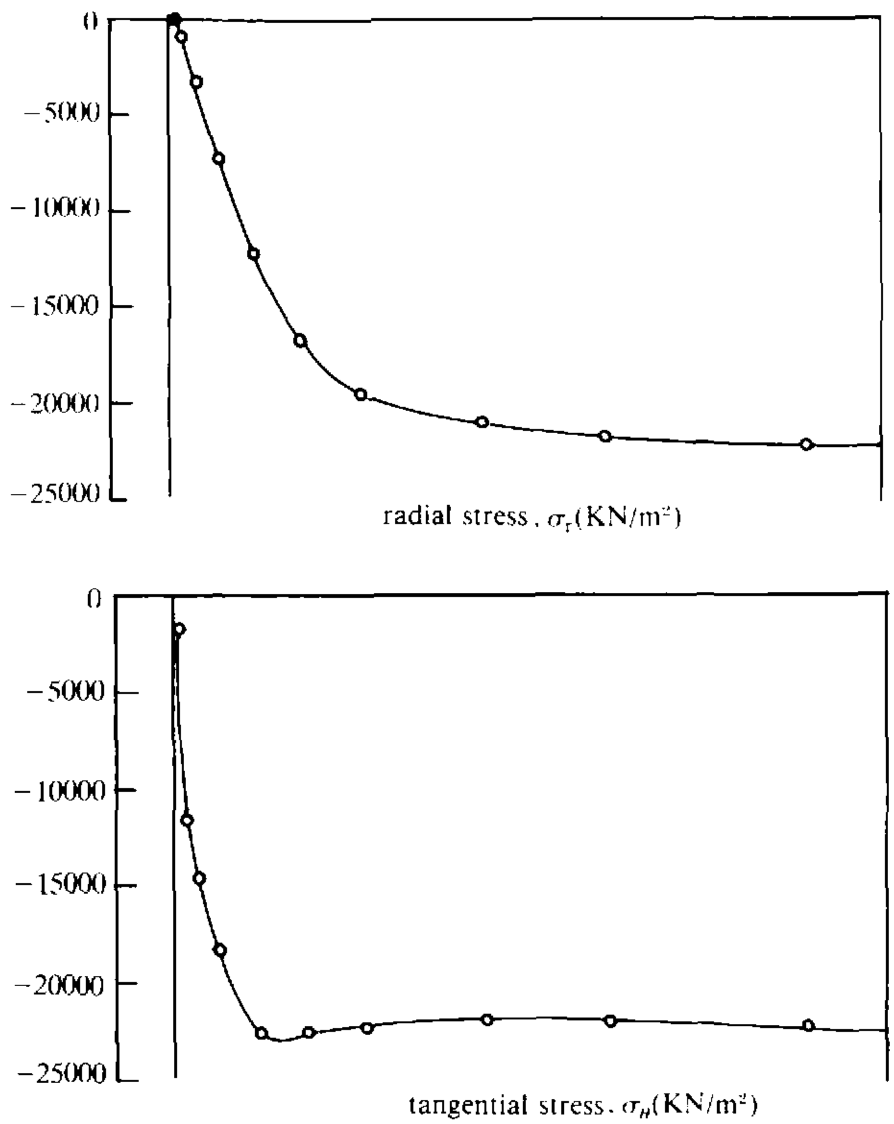


Fig. 8.14 Radial and tangential stress distributions for the problem of Fig. 8.13.

linear field caused by the external applied pressure, superimposed on which is the effect of the cavity presence (the shaded area).

Finally, Fig. 8.14 shows the steady state radial and tangential stress distributions along the line of Gaussian integration points nearest section AB. It is noted that away from the vicinity of the cavity, the hydrostatic condition $\sigma_r = \sigma_\theta$ is reproduced.

8.17 Problems

8.1 Use program VISCOUNT documented in Appendix II, Section A2.2 to solve the thick sphere considered in Problem 7.5 for the viscoplastic case. Employ the same material properties and load increment sizes as used in the elasto-plastic analysis. Assume the fluidity parameter

$\gamma = 0.001$ and flow function $\Phi(F) = F$. Use explicit time integration ($\Theta = 0$) and compare your steady state solutions with the results of Problem 7.5.

8.2 Repeat Problem 8.1 for different limiting time step lengths employing explicit time integration. Take the factor τ, described in Section 8.3, in the range $0.01 \leqslant \tau \leqslant 0.5$. Comment on the accuracy of solution in each case.

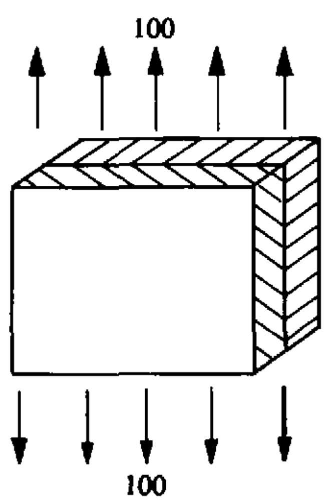
8.3 Repeat Problem 8.1 using the flow functions (8.8) and (8.9). Take the indices M and N in the range 2 to 4. Comment on the solutions.

8.4 Repeat Problem 8.1 using (a) Fully implicit method ($\Theta = 1$) and (b) Implicit trapezoidal rule ($\Theta = \frac{1}{2}$). Comment on the accuracy and computational costs of solution.

8.5 Modify program VISCOUNT to include the strain-hardening law considered in Problem 7.4.

8.6 Undertake all the coding changes required to program VISCOUNT to include the overlay concept described in Section 8.15.

8.7 Test the modified program of Problem 8.6 by employing it in the solution of the uniaxial problem of Fig. 8.15. A constant stress of 100 is applied at time $t = 0$ to the plane stress model shown. Determine the development of strain with time. Verify the numerical solution by noting Figs. 8.4 and 8.5 and hence comparing with the analytical solution of Problem 4.2.



	Overlay 1	Overlay 2
E	1000.0	1000.0
ν	0.0	0.0
t	0.5	0.5
σ_Y	0.0	25.0
H′	100.0	100.0
γ	0.01	0.01

Fig. 8.15 Overlay model example—Problem 8.7.

8.8 In Section 8.2.3 it was stated that large deformation effects could be included, adopting a Lagrangian formulation, by including both the linear and nonlinear terms of the general quadratic relationship between strains and displacements according to (8.19). Details of geometrically nonlinear expressions can be found in Chapters 10 and 11. Modify program VISCOUNT to include such geometrically nonlinear behaviour.

8.9 Employ the modified program of Problem 8.8 to solve the creep buckling problem illustrated in Fig. 8.16. The creep law employed is indicated in Fig. 8.16 and is a particular form of expression (8.9). Using the finite element mesh shown, apply the eccentric load to the cantilever at time, $t = 0$, and employ the implicit time integration algorithm ($\Theta = 1$) to determine the deformation with increasing time. At some stage of the solution process the structure will become unstable due to creep buckling. Carry out the analysis for $\lambda = 1.0, 1.5, 2.0$ and 2.5 and compare the lateral deflection/time relationships with those provided in Ref. 6.

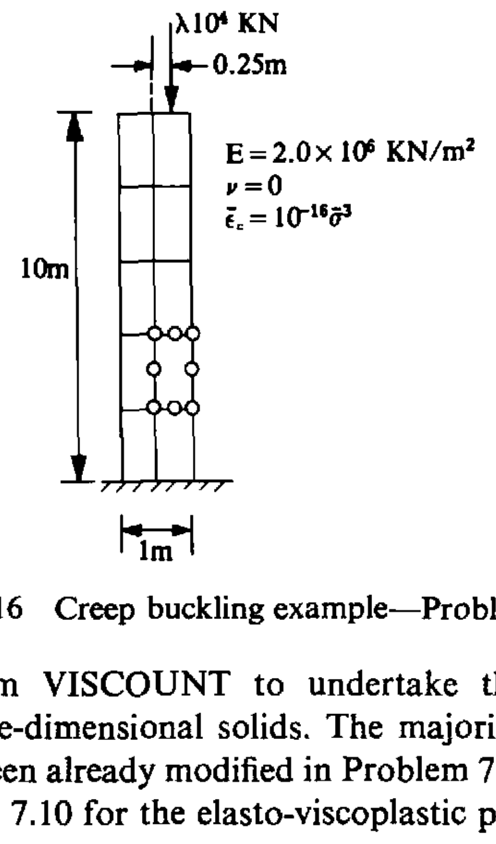


Fig. 8.16 Creep buckling example—Problem 8.9.

8.10 Modify program VISCOUNT to undertake the elasto-viscoplastic solution of three-dimensional solids. The majority of the subroutines required have been already modified in Problem 7.9.

8.11 Repeat Problem 7.10 for the elasto-viscoplastic program VISCOUNT.

8.18 References

1. Olszak, W. and Perzyna, P., Stationary and non-stationary visco-plasticity, In: M. F. Kanninen *et al.* (Eds), *Inelastic Behaviour of Solids*, McGraw-Hill, 1970.
2. Olszak, W. and Perzyna, P., On elasto-viscoplastic soils, In: *Rheology and Soil Mechanics*, IUTAM Symposium, Springer-Verlag, Grenoble, 1966.
3. Perzyna, P., Fundamental problems in visco-plasticity, In: *Recent Advances in Applied Mechanics*, Academic Press, New York, 1966.
4. Zienkiewicz, O. C. and Cormeau, I. C., Visco-plasticity—plasticity and creep in elastic solids—a unified numerical solution approach, *Int. J. Num. Meth. Engng.* **8**, 821–845 (1974).
5. Zienkiewicz, O. C., Owen, D. R. J. and Cormeau, I. C., Analysis of visco-plastic effects in pressure vessels by the finite element method, *Nuclear Engineering & Design*, **28**(2), 278–288 (1974).

6. KANCHI, M. B., ZIENKIEWICZ, O. C. and OWEN, D. R. J., The visco-plastic approach to problems of elasticity and creep involving geometric nonlinear effects. *Int. J. Num. Meth. Engng.* **12**, 169–181 (1978).
7. STRICKLIN, J. A., HAISLER, W. and REISEMANN, W., Evaluation of solution procedures of material and/or geometrically non-linear structural analysis, *AIAA J.* **11**, 292–299 (1973).
8. DINIS, L. M. S. and OWEN, D. R. J., Elastic-viscoplastic analysis of plates by the finite element method, *Computers & Structures*, **8**, 207–215 (1978).
9. CORMEAU, I., Numerical stability in quasistatic elasto-visco-plasticity, *Int. J. Num. Meth. Engng.* **9**, 109–127 (1975).
10. DUWEZ, P., On the Plasticity of Crystals, *Physical Revjew*, **47**(6), 494–501 (1935).
11. ZIENKIEWICZ, O. C., NAYAK, G. C. and OWEN, D. R. J., Composite and overlay models in numerical analysis of elasto-plastic continua, *Int. Symp. Foundations of Plasticity*, Warsaw (1972).
12. OWEN, D. R. J., PRAKASH, A. and ZIENKIEWICZ, O. C., Finite element analysis of non-linear composite materials by use of overlay systems, *Computers & Structures* **4**, 1251–1267 (1974).
13. PANDE, G. N., OWEN, D. R. J and ZIENKIEWICZ, O. C., Overlay models in time-dependent nonlinear material analysis. *Computers & Structures*, **7**, 435–443 (1977).
14. HUGHES, T. J. R. and TAYLOR, R. L., Unconditionally stable algorithms for quasi-static elasto/viscoplastic finite element analysis, *Int. J. Num. Meth. Engng.* (to be published).

Chapter 9
Elasto-plastic Mindlin plate bending analysis

Written in collaboration with M. M. Huq

9.1 Introduction

In Chapter 5 we introduced some elastoplastic Timoshenko beam formulations. In this chapter we introduce some related elasto-plastic Mindlin plate bending formulations.

There are basically three theories which we could use as a basis for elastic plate bending:

(i) *Kirchhoff classical thin plate theory* This theory, which takes no account of transverse shear deformation, is usually favoured by engineers because of its simplicity. It is the plate bending equivalent of Euler–Bernoulli beam theory. Many conforming $C(1)$ and non-conforming $C(0)$ plate elements are available.

(ii) *Mindlin (or Reissner) plate theory* Mindlin and the related Reissner plate theories allow for transverse shear effects. Mindlin plate theory is the plate bending equivalent of Timoshenko beam theory. Several Mindlin plate elements have been presented in the literature and it emerges that the most convenient one is the 'Heterosis' element of Hughes.[1]

(iii) *Full three-dimensional theory* For the greatest accuracy, full three-dimensional theory should be employed. Many 3D hexahedral and tetrahedral elements have been presented. Unfortunately when the aspect ratio of the element is very large as in thin plates, an ill-conditioned stiffness matrix results and roundoff problems predominate. Several schemes for avoiding this difficulty have been presented and undoubtedly an analysis based on this procedure is the most accurate.

Let us now consider the various possibilities for elasto-plastic analysis.

(i) We could use a full 3D analysis with a yield function $F(\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{xz}, \tau_{yz})$.

(ii) In a Mindlin plate formulation we can also use the yield function $F(\sigma_x, \sigma_y, \sigma_z, \tau_{xy}, \tau_{xz}, \tau_{yz})$. It should be noted that σ_z is taken as zero in

Mindlin plates. This approach allows for the spread of plasticity from the extreme fibre over the entire plate thickness. In the evaluation of the internal virtual work integrals we may sample the stresses of the Gauss–Legendre or Lobatto integration points. Alternatively we may divide the plate into layers and use a mid-ordinate rule.

(iii) In a Mindlin or Kirchhoff formulation we can use a yield function $F(\sigma_x, \sigma_y, \tau_{xy})$. In Mindlin plate theory we ignore the effect of τ_{xz} and τ_{yz} on the plastic behaviour. Since, in the absence of inplane forces, the inplane stresses are a maximum at the extreme fibres where the transverse shear stresses are a minimum and the inplane stresses are a minimum at the mid-plane where the transverse shears are a maximum, this is a reasonable assumption. (There is also further evidence to suggest that it is likely to lead to insignificant errors.) This approach also allows for the spread of plasticity over the depth of the plate. In the evaluation of the internal virtual work integrals we may sample the stresses at the Gauss–Legendre or Lobatto integration points. Alternatively we may divide the plate into layers* and use a mid-ordinate rule. This 'layered' approach has been described in Chapter 5 for a Timoshenko beam element and is a very popular method.

(iv) In a Mindlin or Kirchhoff formulation we can adopt in the absence of inplane forces a yield function $F(M_x, M_y, M_{xy})$ which is a function of the bending moments. Here it is assumed that at a point the whole plate section becomes plastic simultaneously. A similar approach was described in Chapter 5 for Timoshenko beam elements.

The elasto-plastic analysis of Mindlin plates is considered in this chapter, where both layered and non-layered approaches are treated in detail.

Finite elements based on Mindlin's assumptions have one important advantage over elements based on classical thin plate theory. Mindlin plate elements require only $C(0)$ continuity of the lateral displacement w and the two independent nodal rotations θ_x and θ_y. However elements based on classical Kirchhoff thin plate theory require $C(1)$ continuity; in other words $\partial w/\partial x$ and $\partial w/\partial y$ as well as w must be continuous across element interfaces. Thus, Mindlin plate elements are simpler to formulate and they have the added advantage of being able to model shear-weak as well as shear-stiff plates. Consequently, if transverse shear deformations are present they are automatically modelled with Mindlin elements.

Recent research[1] indicates that the use of a 'Heterosis' quadrilateral Mindlin plate element with quadratic Lagrangian interpolation for θ_x and θ_y and quadratic Serendipity interpolation for w together with selective integration of the stiffness matrix, gives the best overall performance. It

* These layers are symmetric about the midsurface of the plate in the present formulation.

avoids locking and contains no spurious mechanisms. The Heterosis element is implemented here using a hierarchical formulation described later.

We have already considered elastic Mindlin plate finite element analysis in Chapter 6. Nonlinear Mindlin plate finite element analysis is now considered.

9.2 Equilibrium equations

9.2.1 Three-dimensional equilibrium equations

Let us begin with the equilibrium equations of three-dimensional stress analysis. We will assume that, for convenience, no tractions are present on the boundary Γ_t of the three-dimensional domain Ω. The virtual work equation may be expressed as

$$\int_{\Omega} \{[\delta \boldsymbol{\epsilon}]^T \boldsymbol{\sigma} - [\delta \boldsymbol{u}]^T \boldsymbol{b}\} d\Omega = 0 \tag{9.1}$$

where the vector of virtual displacements in the x, y and z directions is $\delta \boldsymbol{u} = [\delta u,\ \delta v,\ \delta w]^T$, the vector of associated virtual strains is $\delta \boldsymbol{\epsilon} = [\delta \epsilon_x,\ \delta \epsilon_y,\ \delta \epsilon_z,\ \delta \gamma_{xy},\ \delta \gamma_{xz},\ \delta \gamma_{yz}]^T$, the vector of stress is $\boldsymbol{\sigma} = [\sigma_x,\ \sigma_y,\ \sigma_z,\ \tau_{xy},\ \tau_{xz},\ \tau_{yz}]^T$ and the vector of applied body forces is $\boldsymbol{b} = [b_x, b_y, b_z]^T$. Displacements $\boldsymbol{u}$ are prescribed on boundary Γ_u of domain Ω.

The stress–strain relationships for an isotropic material are given as

$$\boldsymbol{D} = a_1 \begin{bmatrix} a_2 & a_3 & a_3 & 0 & 0 & 0 \\ a_3 & a_2 & a_3 & 0 & 0 & 0 \\ a_3 & a_3 & a_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & a_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_4 \end{bmatrix} \tag{9.2}$$

where $a_1 = E/(1+\nu)(1-2\nu)$, $a_2 = 1-\nu$, $a_3 = \nu$ and $a_4 = (1-2\nu)/2$. Note that E is the elastic modulus and ν is Poisson's ratio.

9.2.2 Mindlin plate equilibrium equations

In Mindlin plate theory, the domain of interest Ω is of the special form

$$\Omega = \{(x, y, z) \epsilon R^3 \,|\, z \epsilon [-t/2,\ t/2],\ (x, y) \epsilon A \epsilon R^2\} \tag{9.3}$$

where t is the plate thickness which may be a function of x and y and A is the plate area. The boundary of A is denoted by Γ.

We also make the following set of assumptions:

(i) Normals to the midsurface (i.e., $z = 0$) before deformation remain straight but not necessarily normal to the midsurface after deformation. If θ_x and θ_y are the rotations of the midsurface normal in the xz- and yz- plane respectively, then

$$\mathbf{u} = \begin{bmatrix} u(x, y, z) \\ v(x, y, z) \\ w(x, y, z) \end{bmatrix} = \begin{bmatrix} -z\theta_x(x, y) \\ -z\theta_y(x, y) \\ w(x, y) \end{bmatrix} \tag{9.4}$$

The sign convention is illustrated in Fig. (9.1). Right hand rotations $\bar{\theta}_x$ and $\bar{\theta}_y$ are defined by the expression

$$\begin{bmatrix} \theta_x \\ \theta_y \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \bar{\theta}_x \\ \bar{\theta}_y \end{bmatrix}. \tag{9.5}$$

It is usually more convenient to develop the theory in terms of θ_x and θ_y rather than $\bar{\theta}_x$ and $\bar{\theta}_y$ since the resulting algebra is greatly simplified.

(ii) The normal stress σ_z is assumed equal to zero. The virtual work statement may be expressed as

$$\int_\Omega [\delta\boldsymbol{\epsilon}']^T \boldsymbol{\sigma}' \, d\Omega - \int_\Omega [\delta \boldsymbol{u}]^T \boldsymbol{b} \, d\Omega = 0 \tag{9.6*}$$

in which

$$[\delta\boldsymbol{\epsilon}'] = [\delta\epsilon_x, \delta\epsilon_y, \delta\gamma_{xy} \mid \delta\gamma_{xz}, \delta\gamma_{yz}]^T = [(\delta\boldsymbol{\epsilon}_f)^T, (\delta\boldsymbol{\epsilon}_s)^T]^T$$

and

$$\boldsymbol{\sigma}' = [\sigma_x, \sigma_y, \sigma_z \mid \tau_{xz}, \tau_{yz}]^T = [(\boldsymbol{\sigma}_f)^T, (\boldsymbol{\sigma}_s)^T]^T.$$

Note that

$$\delta\boldsymbol{\epsilon}_f = z\left[-\frac{\partial(\delta\theta_x)}{\partial x}, \; -\frac{\partial(\delta\theta_y)}{\partial y}, \; -\left(\frac{\partial(\delta\theta_x)}{\partial y} + \frac{\partial(\delta\theta_y)}{\partial x}\right)\right]^T = z\,\delta\hat{\boldsymbol{\epsilon}}_f \tag{9.7†}$$

* In Mindlin plate theory a reduced form of the constitutive relations is obtained by making $\sigma_z = 0$ and subsequently eliminating ϵ_z. Thus

$$\boldsymbol{\sigma}' = \boldsymbol{D}'\boldsymbol{\epsilon}'$$

where for elastic isotropic situations

$$\boldsymbol{D}' = \frac{E}{(1-\nu^2)} \left[\begin{array}{ccc|cc} 1 & \nu & 0 & 0 & 0 \\ \nu & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{(1-\nu)}{2} & 0 & 0 \\ \hline 0 & 0 & 0 & \frac{(1-\nu)}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{(1-\nu)}{2} \end{array}\right] = \begin{bmatrix} \boldsymbol{D}_f' & \mathbf{0} \\ \mathbf{0} & \boldsymbol{D}_s' \end{bmatrix}$$

† Terms symbolised thus (^) denote quantities integrated over the thickness.

and

$$\delta\boldsymbol{\epsilon}_s = \left[\frac{\partial(\delta w)}{\partial x} - \delta\theta_x, \quad \frac{\partial(\delta w)}{\partial y} - \delta\theta_y\right]^T = \delta\hat{\boldsymbol{\epsilon}}_s. \tag{9.8}$$

Using (9.7) and (9.8) we find that (9.6) can be rewritten as

$$\int_A \int_{-t/2}^{t/2} [z(\delta\hat{\boldsymbol{\epsilon}}_f)^T \boldsymbol{\sigma}_f + (\delta\hat{\boldsymbol{\epsilon}}_s)^T \boldsymbol{\sigma}_s - (\delta\boldsymbol{u})^T \boldsymbol{b}]dz\, dA = 0 \tag{9.9}$$

This equation is adopted in the layered approach. After integration over the thickness of the plate (9.9) can be rewritten in the form

$$\int_A [(\delta\hat{\boldsymbol{\epsilon}}_f)^T \hat{\boldsymbol{\sigma}}_f + (\delta\boldsymbol{\epsilon}_s)^T \hat{\boldsymbol{\sigma}}_s - (\delta\boldsymbol{u})^T \hat{\boldsymbol{b}}]dA = 0 \tag{9.10}$$

where

$$\hat{\boldsymbol{\sigma}}_f = \int_{-t/2}^{t/2} z\,\boldsymbol{\sigma}_f\, dz$$

$$\hat{\boldsymbol{\sigma}}_s = \int_{-t/2}^{t/2} \boldsymbol{\sigma}_s\, dz$$

and

$$\hat{\boldsymbol{b}} = \int_{-t/2}^{t/2} \boldsymbol{b}\, dz.$$

We interpret $\hat{\boldsymbol{\sigma}}_f = [M_x, M_y, M_{xy}]^T$ as the bending moments and $\hat{\boldsymbol{\sigma}}_s = [Q_x, Q_y]^T$ as the shear force. Usually we take $\hat{\boldsymbol{b}} = [q, 0, 0]^T$ in which q is the lateral distributed loading acting on the plate. We use (9.10) in the non-layered plate formulation.

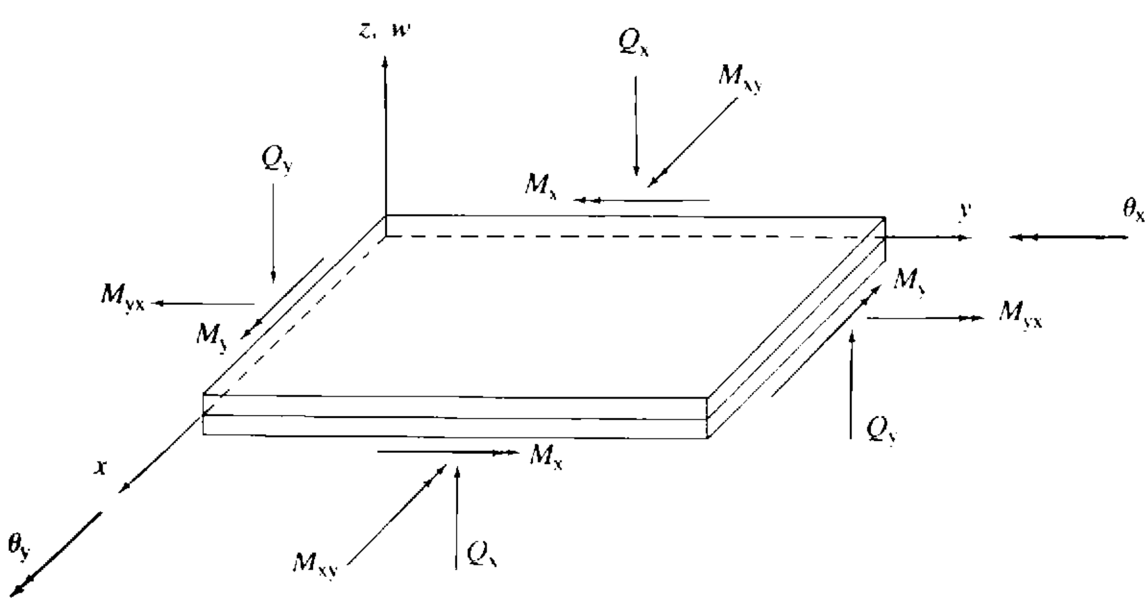


Fig. 9.1 Sign convention for Mindlin plate theory.

9.3 Discretisation

9.3.1 Standard representation

If we adopt a standard $C(0)$ finite element representation then the displacements can be written as

$$\boldsymbol{u} = \sum_{i=1}^{n} \boldsymbol{N}_i \boldsymbol{d}_i \tag{9.11}$$

in which the shape function matrix is $\boldsymbol{N}_i = N_i \boldsymbol{I}_3$ and the vector of nodal values $\boldsymbol{d}_i = [w_i, \theta_{xi}, \theta_{yi}]^T$.

The flexural strain displacement equations are given as

$$\delta\hat{\boldsymbol{\epsilon}}_f = \sum_{i=1}^{n} \boldsymbol{B}_{fi}\, \delta\boldsymbol{d}_i \tag{9.12}$$

in which

$$\boldsymbol{B}_{fi} = \begin{bmatrix} 0 & -\dfrac{\partial N_i}{\partial x} & 0 \\ 0 & 0 & -\dfrac{\partial N_i}{\partial y} \\ 0 & -\dfrac{\partial N_i}{\partial y} & -\dfrac{\partial N_i}{\partial x} \end{bmatrix}$$

The shear strain displacement equations have the form

$$\delta\hat{\boldsymbol{\epsilon}}_s = \sum_{i=1}^{n} \boldsymbol{B}_{si}\, \delta\boldsymbol{d}_i \tag{9.13}$$

in which

$$\boldsymbol{B}_{si} = \begin{bmatrix} \dfrac{\partial N_i}{\partial x} & -N_i & 0 \\ \dfrac{\partial N_i}{\partial y} & 0 & -N_i \end{bmatrix}$$

If we substitute (9.11)–(9.13) in (9.9) we obtain the expression

$$\sum_{i=1}^{n} [\delta\boldsymbol{d}_i]^T \left\{ \int_A \int_{-t/2}^{t/2} [\boldsymbol{B}_{fi}]^T \boldsymbol{\sigma}_f z + [\boldsymbol{B}_{si}]^T \boldsymbol{\sigma}_s - [\boldsymbol{N}_i]^T \boldsymbol{b}] dz\, dA \right\} = 0. \tag{9.14}$$

Since (9.14) must be true for any set of virtual displacements we obtain the expression

$$\int_A \int_{-t/2}^{t/2} [[\boldsymbol{B}_{fi}]^T \boldsymbol{\sigma}_f z + [\boldsymbol{B}_{si}]^T \boldsymbol{\sigma}_s - [N_i]^T \boldsymbol{b}] dz\, dA = 0 \tag{9.15}$$

or

$$\boldsymbol{\psi}_i(\boldsymbol{d}) = 0.$$

We use (9.15) in the layered approach. If we integrate the terms in square brackets over the thickness of the plate then we obtain the following equation

$$\int_A [[\boldsymbol{B}_{fi}]^T \hat{\boldsymbol{\sigma}}_f + [\boldsymbol{B}_{si}]^T \hat{\boldsymbol{\sigma}}_s - [N_i]^T \hat{\boldsymbol{b}}] dA = 0 \tag{9.16}$$

or

$$\boldsymbol{\psi}_i(\boldsymbol{d}) = 0.$$

Equation (9.16) is used in the nonlayered approach.

Note that we obtain equations for the residual force vector $\boldsymbol{\psi}_i(\boldsymbol{d})$ for every node in the finite element discretisation. When the stresses are nonlinear then both (9.15) and (9.16) are sets of nonlinear simultaneous equations.

Contributions to the residual force vector $\boldsymbol{\psi} = [\boldsymbol{\psi}_1{}^T, \ldots, \boldsymbol{\psi}_n{}^T]^T$ may be evaluated at the element level and then assembled to form $\boldsymbol{\psi}$. We may use any standard $C(0)$ two-dimensional isoparametric element. Several elements have been presented in the literature and it emerges that the most convenient one is the 8/9 node 'Heterosis' element of Hughes.[1] In the programs described later we use 4, 8 and 9-noded isoparametric quadrilateral elements (see Chapter 6), as well as the Heterosis element. Selective integration is adopted and this will be described later.

9.3.2 Hierarchical formulation of the Heterosis element

In the implementation of the Heterosis and the 9-node element a hierarchical formulation is adopted. The first 8 shape functions are borrowed from the 8-noded Serendipity element and the shape function for the central 9th node is the bubble function

$$N_9{}^{(e)}(\xi, \eta) = (1-\xi^2)(1-\eta^2) \tag{9.17}$$

which is already available from the quadratic Lagrangian element. This means that all variables associated with the central node are hierarchical in nature. In other words, they are departures from the interpolated Serendipity values. The hierarchical representation can be used for geometrical representation as well as for interpolating displacements.

In order to implement the heterosis element we adopt a hierarchical formulation either by adding a stiff spring (large number) to the leading diagonal term of the stiffness matrix associated with the lateral displacement parameter for node 9, or by prescribing displacement at this centre node to zero. This has the effect of forcing w to behave as though it was represented by Serendipity quadratic shape functions. Thus the desired effect is achieved.

It is worth noting that if no spring is added the element obtained is identical to the 9-noded Lagrangian element provided that care is taken in evaluating the consistent nodal forces. Furthermore if stiff springs are added to all the terms of the leading diagonal associated with node 9, then the element reverts to a Serendipity 8-noded element.

For convenience, in the present case, when representing the geometry of the heterosis element, the x and y coordinate departures from the interpolated Serendipity values are taken as equal to zero. In other words, as Serendipity geometrical representation is adopted this distinction is only of importance when elements with curved boundaries are present. (N.B. This is automatically taken care of by a modified version of Subroutine NODEXY described in Section 6.4.1).

9.4 Solution of nonlinear equations

9.4.1 Plasticity in layered plates

For Mindlin plates we may assume that the yield function F is a function of σ_f, the direct stresses associated with flexure, but not of the transverse shear stresses σ_s. The yield function F is also a function of the hardening parameter, H. When yielding occurs at some point, it is assumed that, unless unloading takes place, the stresses always remain on the yield surface so that

$$F(\sigma_f, H) = 0 \tag{9.18}$$

Thus the incremental stress–strain relationship is given as

$$\begin{bmatrix} d\boldsymbol{\sigma}_f \\ d\boldsymbol{\sigma}_s \end{bmatrix} = \begin{bmatrix} (\boldsymbol{D}_{ep}')_f & \mathbf{0} \\ \mathbf{0} & \boldsymbol{D}_s' \end{bmatrix} \begin{bmatrix} d\boldsymbol{\epsilon}_f \\ d\boldsymbol{\epsilon}_s \end{bmatrix} \tag{9.19}$$

or

$$d\boldsymbol{\sigma}' = \boldsymbol{D}_{ep}' d\boldsymbol{\epsilon}'$$

in which $(\boldsymbol{D}_{ep}')_f$ is identical to $\boldsymbol{D}_{ep}$ given in Chapter 7 for the elasto-plastic plane stress problem. Note that $\boldsymbol{D}_s'$ always remains elastic. Recall from equation (7.47) that

$$(\boldsymbol{D}_{ep}')_f = \boldsymbol{D}_f' - \frac{\boldsymbol{d}_D\, \boldsymbol{d}_D^T}{A + \boldsymbol{d}_D^T \boldsymbol{a}'} \tag{9.20}$$

where

$$\boldsymbol{a}' = \left[\frac{\partial F}{\partial \sigma_x}, \frac{\partial F}{\partial \sigma_y}, \frac{\partial F}{\partial \tau_{xy}}\right]^T$$

$$d_D = D_f' a'$$

$$A = -\frac{1}{\lambda} \frac{\partial F}{\partial H} dH$$

in which λ is the proportionality constant. Here we cater for Von Mises and Tresca materials only. We can thus use a slightly modified version of the coding described in Chapter 7 when evaluating $(D_{ep}')_f$ and when testing for yielding etc.

9.4.2 Solution of the nonlinear equilibrium equations for layered plates

The incremental equilibrium equations for the plate can be written at some stage in the solution (i.e., at an iteration during a load increment) as

$$\psi(d^p) + K_T(d^p)\Delta d^p = 0 \tag{9.21}$$

where ψ is obtained from (9.15) and $K_T(d^p)$ is the tangential stiffness matrix which may be approximated as

$$K_T(d^p) = \int_A \int_{-t/2}^{t/2} \{[B_f]^T [D_{ep}']_f B_f + [B_s]^T D_s' B_s\} dz\, dA. \tag{9.22}$$

Since $[D_{ep}']_f$ is a function of z we may employ a numerical integration technique to evaluate the integral over the thickness of the plate. Here, we divide the plate into layers and use a mid-ordinate rule as described in Chapter 5 for the Timoshenko beam. We use a similar method to evaluate $\psi(d^p)$. Thus we have

$$K_T(d^p) = \int_A \{[B_f]^T [\hat{D}_{ep}]_f B_f + [B_s]^T \hat{D}_s B_s]\} dA \tag{9.23}$$

where

$$[\hat{D}_{ep}]_f = \int_{-t/2}^{t/2} [D_{ep}']_f\, dz$$

and

$$\hat{D}_s = \int_{-t/2}^{t/2} D_s'\, dz.$$

We now use the standard procedure to solve (9.21). Instead of using $K_T(d^p)$ we may use some previously calculated value of K_T just as in the other applications.

9.4.3 Plasticity in nonlayered plates

In Chapter 5 we considered the elasto-plastic nonlayered analysis of Timoshenko beams in which we assumed that when the bending moment reaches the yield moment M_0, the whole cross-section of the beam becomes plastic instantaneously. We noted that this is a convenient fiction as in reality there is always a gradual spread of plasticity over the depth of the beam. In elasto-plastic nonlayered Mindlin plate analysis we make a similar approximation. Here we assume that the yield function $\hat{F}$ is expressed as a function of the bending moments $\hat{\boldsymbol{\sigma}}_f$, but not of the shear forces $\hat{\boldsymbol{\sigma}}_s$. The yield function is also assumed to be a function of a hardening parameter $\hat{H}$. During yield it is assumed that the stress resultants $\hat{\boldsymbol{\sigma}}_f$ must remain on the yield surface so that

$$\hat{F}(\hat{\boldsymbol{\sigma}}_f, \hat{H}) = 0 \tag{9.24}$$

where for the Tresca and Von Mises materials under consideration

$$\hat{F}(\hat{\boldsymbol{\sigma}}_f, \hat{H}) = \int_{-t/2}^{t/2} F(\boldsymbol{\sigma}_f, H)dz. \tag{9.25}$$

Therefore, although $\hat{F}$ replaces F, (M_x, M_y, M_{xy}) replace $(\sigma_x, \sigma_y, \tau_{xy})$ and $M_0 = \sigma_0 t^2/4$ replaces σ_0, everything else remains unchanged and we can again make use of the coding given in Chapter 7.

The incremental stress–strain resultant relationships are given as

$$\begin{bmatrix} d\hat{\boldsymbol{\sigma}}_f \\ d\hat{\boldsymbol{\sigma}}_s \end{bmatrix} = \begin{bmatrix} [\hat{\boldsymbol{D}}_{ep}]_f & \boldsymbol{0} \\ \boldsymbol{0} & \hat{\boldsymbol{D}}_s \end{bmatrix} \begin{bmatrix} d\hat{\boldsymbol{\epsilon}}_f \\ d\hat{\boldsymbol{\epsilon}}_s \end{bmatrix} \tag{9.26}$$

in which

$$[\hat{\boldsymbol{D}}_{ep}]_f = \hat{\boldsymbol{D}}_f - \frac{\hat{\boldsymbol{d}}_D\, \hat{\boldsymbol{d}}_D{}^T}{\hat{A} + \hat{\boldsymbol{d}}_D{}^T\, \hat{\boldsymbol{a}}} \tag{9.27}$$

in which

$$\hat{\boldsymbol{a}} = \left[\frac{\partial \hat{F}}{\partial M_x}, \frac{\partial \hat{F}}{\partial M_y}, \frac{\partial \hat{F}}{\partial M_{xy}}\right]^T$$

$$\hat{\boldsymbol{d}}_D = \hat{\boldsymbol{D}}_f \hat{\boldsymbol{a}}$$

$$\hat{A} = -\frac{1}{\lambda} \frac{\partial \hat{F}}{\partial \hat{H}} d\hat{H}$$

and

$$\hat{\boldsymbol{D}}_f = \int_{-t/2}^{t/2} \boldsymbol{D}_f{}' z\, dz.$$

Note also that

$$\hat{\boldsymbol{D}}_s = \int_{-t/2}^{t/2} \boldsymbol{D}_s{}' dz.$$

9.4.4 Solution of nonlinear equilibrium equations for nonlayered Mindlin plates

For the nonlayered plates the equilibrium equations are identical to (9.21). Here, however, the tangential stiffness matrix is given as

$$K_T = \int_A \{[B_f]^T [\hat{D}_{ep}]_f B_f + [B_s]^T \hat{D}_s B_s\} dA. \tag{9.28}$$

Apart from this modification the solution procedure is unchanged.

9.4.5 Summary of solution procedures

The solution procedures for elasto-plastic Mindlin plate analysis are summarised in Tables 9.1–9.3. The overall process is given in Table 9.1. The iteration loop is shown for the nonlayered and layered plates in Tables 9.2 and 9.3 respectively.

Table 9.1 Equation solving technique for layered and nonlayered Mindlin plates

1	Begin new load increment, $f = f + \Delta f$.
2	Set Δf equal to the current load increment vector.
3	Set d^0 equal to $\mathbf{0}$ for the first increment or equal to the total displacement vector at the end of the last load increment.
4	Set ψ^0 equal to the residual force vector at the end of the last increment or equal to $\mathbf{0}$ for the first load increment.
5	Set $\psi^0 = \psi^0 + \Delta f$.
6	Solve $\Delta d^0 = -[K_T]^{-1} \psi^0$. Use old or updated value K_T.
7	Set $d^1 = d^0 + \Delta d^0$.
8	Evaluate $\psi^1(d^1)$.
9	If solution has converged go to 11; otherwise continue.
10	Iterate until solution has converged.
11	If this is not the last increment go to 1; otherwise stop.

Table 9.2 The iteration loop for elasto-plastic nonlayered Mindlin plates.

1	Set iteration number $i = 1$.
2	Solve $\Delta d^i = -[K_T]^{-1} \psi^i$. Use old or updated K_T.
3	Set $d^{i+1} = d^i + \Delta d^i$.
4	For each Gauss point, evaluate the increments in strain resultants $$\Delta \hat{\epsilon}_f{}^i = B_f \Delta d^i$$ $$\Delta \hat{\epsilon}_s{}^i = B_s \Delta d^i.$$

Table 9.2—*continued*

5	Using the elastic rigidities estimate, at each Gauss point, the increments in stress resultants and hence the total stress resultants $$\Delta\hat{\boldsymbol{\sigma}}_f^{\,i} = \hat{\boldsymbol{D}}_f \Delta\hat{\boldsymbol{\epsilon}}_f^{\,i} \quad \text{hence} \quad \hat{\boldsymbol{\sigma}}_f^{\,i+1} = \hat{\boldsymbol{\sigma}}_f^{\,i} + \Delta\hat{\boldsymbol{\sigma}}_f^{\,i}$$ $$\Delta\hat{\boldsymbol{\sigma}}_s^{\,i} = \hat{\boldsymbol{D}}_s \Delta\hat{\boldsymbol{\epsilon}}_s^{\,i} \quad \text{hence} \quad \hat{\boldsymbol{\sigma}}_s^{\,i+1} = \hat{\boldsymbol{\sigma}}_s^{\,i} + \Delta\hat{\boldsymbol{\sigma}}_s^{\,i}.$$
6	At each Gauss point, depending on the states of $\hat{\boldsymbol{\sigma}}_f^{\,i}$ and $\hat{\boldsymbol{\sigma}}_f^{\,i+1}$, adjust $\hat{\boldsymbol{\sigma}}_f^{\,i+1}$ to satisfy the yield criterion and preserve the normality condition.
7	Evaluate the residual force vector $$\boldsymbol{\psi}^{i+1} = \int_A \{[\boldsymbol{B}_f]^T \hat{\boldsymbol{\sigma}}_f + [\boldsymbol{B}_s]^T \hat{\boldsymbol{\sigma}}_s\} dA - \boldsymbol{f}.$$
8	If the solution has converged, continue, otherwise set $i = i+1$ and go to 2.
9	Move to next load increment.

Table 9.3 The iteration loop for elasto-plastic layered Mindlin plates.

1	Set iteration number $i = 1$.
2	Solve $\Delta\boldsymbol{d}^i = -[\boldsymbol{K}_T]^{-1}\boldsymbol{\psi}^i$. Use old or updated $\boldsymbol{K}_T$.
3	Set $\boldsymbol{d}^{i+1} = \boldsymbol{d}^i + \Delta\boldsymbol{d}^i$.
4	For each Gauss point in each layer evaluate the increment in strain $$\Delta\boldsymbol{\epsilon}_f^{\,i} = z\boldsymbol{B}_f \Delta\boldsymbol{d}^i$$ $$\Delta\boldsymbol{\epsilon}_s^{\,i} = \boldsymbol{B}_s \Delta\boldsymbol{d}^i.$$
5	Estimate the increments in stress at each Gauss point in each layer using the elastic stress–strain matrix. Hence evaluate the total stress value. $$\Delta\boldsymbol{\sigma}_f^{\,i} = \mathbf{D}_f'\Delta\boldsymbol{\epsilon}_f', \qquad \boldsymbol{\sigma}_f^{\,i+1} = \boldsymbol{\sigma}_f^{\,i} + \Delta\boldsymbol{\sigma}_f^{\,i}$$ $$\Delta\boldsymbol{\sigma}_s^{\,i} = \mathbf{D}_s'\Delta\boldsymbol{\epsilon}_s', \qquad \boldsymbol{\sigma}_s^{\,i+1} = \boldsymbol{\sigma}_s^{\,i} + \Delta\boldsymbol{\sigma}_s^{\,i}.$$
6	Depending on the states of $\boldsymbol{\sigma}_f^{\,i}$ and $\boldsymbol{\sigma}_f^{\,i+1}$, adjust $\boldsymbol{\sigma}_f^{\,i+1}$ to satisfy the yield criterion and preserve the normality condition.
7	Evaluate the stress resultants $\hat{\boldsymbol{\sigma}}_f^{\,i+1}$ and $\hat{\boldsymbol{\sigma}}_s^{\,i+1}$ at each Gauss point.
8	Evaluate the residual force vector $$\boldsymbol{\psi}^{i+1} = \int_A \{[\boldsymbol{B}_f]^T \hat{\boldsymbol{\sigma}}_f + [\boldsymbol{B}_s]^T \hat{\boldsymbol{\sigma}}_s\} dA - \boldsymbol{f}.$$
9	If the solution has converged continue, otherwise set $i = i+1$ and go to 2.
10	Move to next load increment.

In this application we recommended the following convergence criteria. Let

$$E_\delta = \frac{\left[\sum_j (\Delta\delta_j^{(i)})^2\right]^{1/2}}{\left[\sum_j (\delta_j^{(i+1)})^2\right]^{1/2}} \tag{9.29}$$

where δ_j may equal w_j, θ_{xj} or θ_{yj}. We take in any combination

$$E_w, E_{\theta x}, E_{\theta y}, (E_w + E_{\theta x} + E_{\theta y}) \leqslant \text{TOLER} \tag{9.30}$$

where TOLER is a specified tolerance. We can also take the residual force equivalents of w_j, θ_{xj} or θ_{yj} in (9.29) and (9.30).

9.5 Software for the non-layered approach

9.5.1 Overall program structure

The overall program structure for the elasto-plastic Mindlin plate bending analysis program MINDLIN using a nonlayered approach is given in Fig. 9.2.

The dimensions given in subroutine FEMP agree with those given in subroutine DIMMP and limit the program to the following maximum size problems in the present form

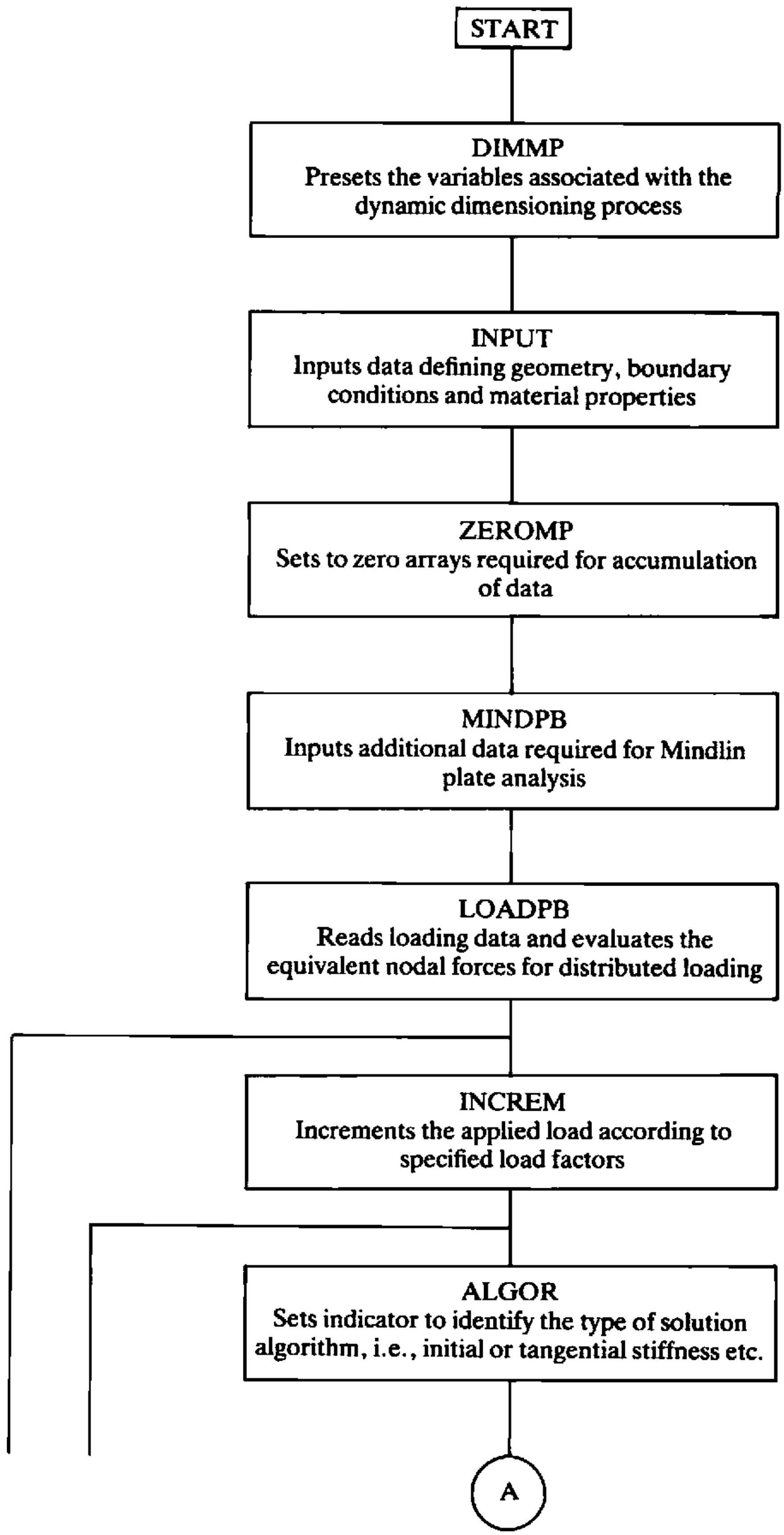
MELEM	– maximum number of elements	= 25
MEVAB	– maximum number of variables per element	= 27
MFRON	– maximum front width	= 40
MMATS	– maximum number of material sets	= 10
MPOIN	– maximum number of nodal points	= 80
MTOTV	– maximum total number of degrees of freedom	= 240
MVFIX	– maximum number of prescribed boundary nodes	= 40

To modify these values the DIMENSION statement in FEMP and the appropriate statements in DIMMP should be *carefully changed and checked*. All new routines are now documented and these include: FEMP, CONVMP, DIMMP, FLOWMP, GRADMP, INVMP, MINDPB, OUTMP, SFR2,* RESMP, STIFMP, STRMP, SUBMP, VZERO and ZEROMP. The other routines, which have been described earlier, include ALGOR, BMATPB, CHECK1,† CHECK2, ECHO, FRONT, INCREM, INPUT, JACOB2, MODPB and NODEXY.*

The files which are used in the program are 5 (cardreader), 6 (lineprinter) and 1, 2, 3, 4, 8 (scratch files).

* Note we include the modified versions of SFR2 and NODEXY to allow for hierarchical representation.

† We include a very slightly modified version of CHECK 1. Note also that for 4-node Mindlin plate elements, GAUSSQ is modified to allow for a single point Gauss rule. See Section 6.4.2.


START
DIMMP
Presets the variables associated with the dynamic dimensioning process
INPUT
Inputs data defining geometry, boundary conditions and material properties
ZEROMP
Sets to zero arrays required for accumulation of data
MINDPB
Inputs additional data required for Mindlin plate analysis
LOADPB
Reads loading data and evaluates the equivalent nodal forces for distributed loading
INCREM
Increments the applied load according to specified load factors
ALGOR
Sets indicator to identify the type of solution algorithm, i.e., initial or tangential stiffness etc.
A

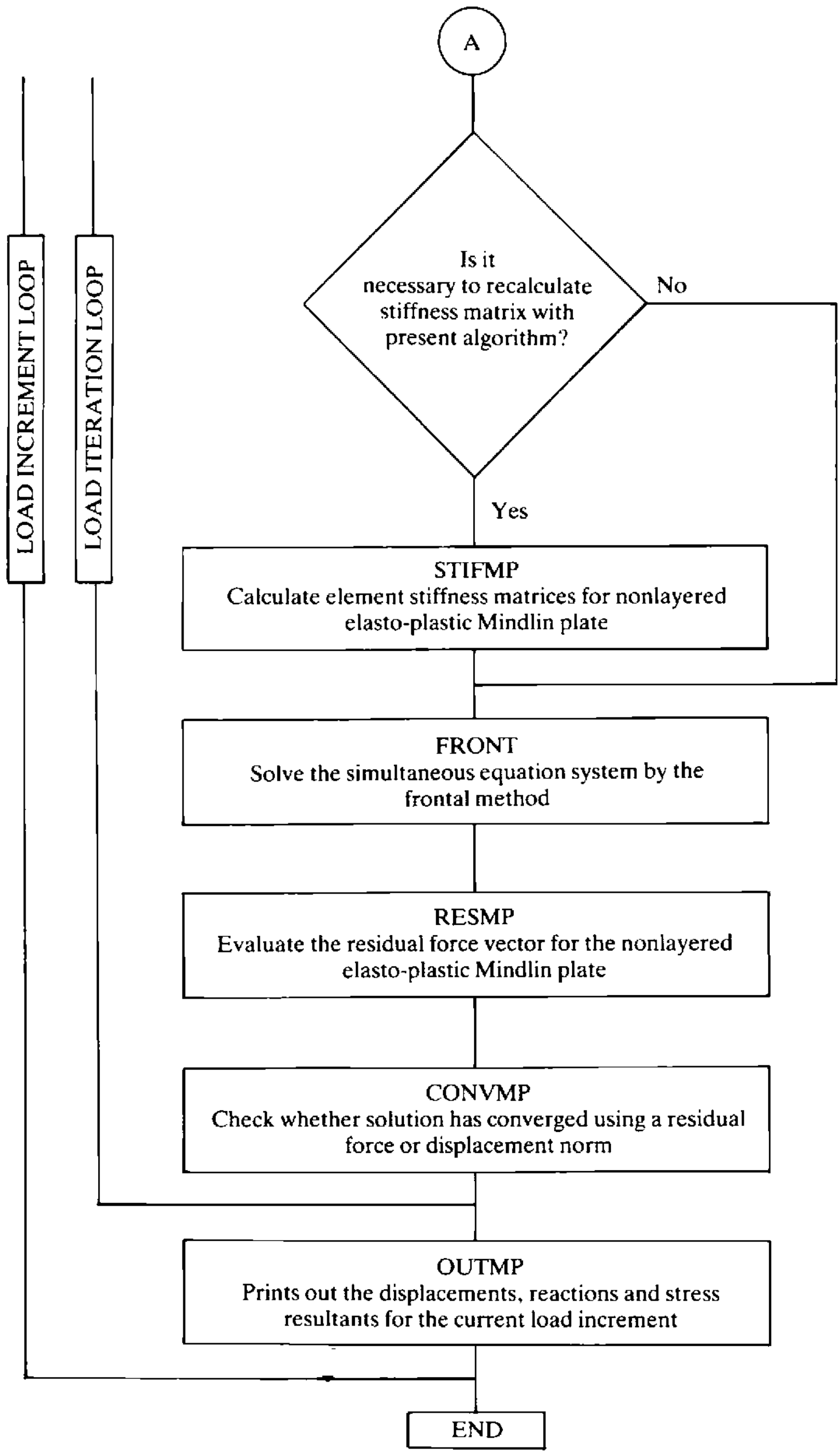


Fig. 9.2 Overall structure of program MINDLIN.

9.5.2 Subroutine FEMP

This routine controls the calling sequence of all of the other main routines as indicated in Fig. 9.2.

```
      PROGRAM FEMP(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
     .TAPE1,TAPE2,TAPE3,TAPE4,TAPE8,TAPE9)
C***********************************************************************
C
C*** ELASTO-PLASTIC ANALYSIS OF NON-LAYERED MINDLIN PLATES USING
C*** 4-,8- , 9-NODED OR HETEROSIS ISOPARAMETRIC QUADRILATERALS
C
C***********************************************************************
      DIMENSION ASDIS(240),COORD(80,2),EFFST(225),ELOAD(25,27),
     .          EPSTN(225),ESTIF(27,27),
     .          EQRHS(10),EQUAT(40,10),FIXED(240),
     .          IFFIX(240),GLOAD(40),GSTIF(860),LNODS(25,9),LOCEL(27),
     .          MATNO(25),NACVA(40),NAMEV(10),NCDIS(4),NCRES(4),
     .          NDEST(27),NDFRO(25),NOFIX(40),NOUTP(2),NPIVO(10),
     .          POSGP(4),PRESC(40,3),PROPS(10,8),REFOR(240),
     .          RLOAD(25,27),STRSG(5,225),TOFOR(240),
     .          TDISP(240),TLOAD(25,27),TREAC(40,3),VECRV(40),
     .          WEIGP(4)
C
C*** PRESET VARIABLES ASSOCIATED WITH DYNAMIC DIMENSIONS
C
      CALL       DIMMP     (MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,
     .                      MSTIF,MTOTG,MTOTV,MVFIX,NDIME,NDOFN,
     .                      NPROP,NSTRE)
C
C*** CALL THE SUBROUTINE WHICH READS MOST OF THE PROBLEM DATA
C
      CALL       INPUT     (COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,
     .                      MFRON,MMATS,MPOIN,MTOTV,MVFIX,NALGO,
     .                      NCRIT,NDFRO,NDIME,NDOFN,NELEM,NEVAB,
     .                      NGAUS,NLAPS,NINCS,NMATS,NNODE,NOFIX,
     .                      NPOIN,NPROP,NSTRE,NSTR1,NSWIT,NTOTG,
     .                      NTOTV,NTYPE,NVFIX,POSGP,PRESC,PROPS,
     .                      WEIGP)
C
C*** INITIALIZE ARRAYS TO ZERO
C
      CALL       ZEROMP    (EFFST,ELOAD,EPSTN,MELEM,MEVAB,MTOTG,
     .                      MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NGAUS,
     .                      NTOTG,NTOTV,NVFIX,STRSG,TDISP,TFACT,
     .                      TLOAD,TREAC)
C
C***
C
      CALL       MINDPB    (IFDIS,IFFIX,IFRES,LNODS,MELEM,MTOTV,
     .                      NCDIS,NCRES,NELEM,NTYPE)
C
C
C
C*** COMPUTE LOAD AFTER READING RELEVANT EXTRA DATA
C
      CALL       LOADPB    (COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,
     .                      NELEM,NEVAB,NGAUS,NNODE,NPOIN,PROPS,
     .                      RLOAD)
C
C*** LOOP OVER EACH INCREMENT
C
```

```
      DO 70 IINCS=1,NINCS
C
C*** READ DATA FOR CURRENT INCREMENT
C
      CALL        INCREM     (ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,
     .                        MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NOUTP,
     .                        NOFIX,NTOTV,NVFIX,PRESC,RLOAD,TFACT,
     .                        TLOAD,TOLER)
C
C*** LOOP OVER EACH ITERATION
C
      DO 90 IITER=1,MITER
C
C*** CALL ROUTINE WHICH SELECTS SOLUTION ALGORITHM VARIABLE KRESL
C
      CALL        ALGOR      (FIXED,IINCS,IITER,KRESL,MTOTV,NALGO,
     .                        NTOTV)
C
C*** CHECK WHETHER A NEW EVALUATION OF THE STIFFNESS MATRICES IS NEEDED
C
      IF(KRESL.EQ.1)
     .CALL        STIFMP     (COORD,EPSTN,IINCS,LNODS,MATNO,MELEM,
     .                        MEVAB,MMATS,MPOIN,MTOTG,NCRIT,NELEM,
     .                        NEVAB,NGAUS,NNODE,PROPS,STRSG)
C
C*** SOLVE EQUATIONS
C
      CALL        FRONT      (ASDIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,
     .                        IFFIX,IINCS,IITER,GLOAD,GSTIF,KRESL,
     .                        LNODS,LOCEL,MBUFA,MELEM,MEVAB,MFRON,
     .                        MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,
     .                        NDOFN,NELEM,NEVAB,NNODE,NOFIX,NPIVO,
     .                        NPOIN,NTOTV,TDISP,TLOAD,TREAC,VECRV)
C
C*** CALCULATE RESIDUAL FORCES
C
      CALL        RESMP      (ASDIS,COORD,EFFST,ELOAD,EPSTN,LNODS,
     .                        MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,
     .                        NCRIT,NELEM,NEVAB,NGAUS,NNODE,PROPS,
     .                        STRSG)
C
C*** CHECK FOR CONVERGENCE
C
      CALL        CONVMP     (ASDIS,ELOAD,IITER,IFDIS,IFRES,LNODS,
     .                        MELEM,MEVAB,MTOTV,NCHEK,NCDIS,NCRES,
     .                        NDOFN,NELEM,NEVAB,NNODE,NPOIN,NTOTV,
     .                        REFOR,TOFOR,TDISP,TLOAD,TOLER)
C
C*** OUTPUT RESULTS IF REQUIRED

C

      IF(IITER.EQ.1.AND.NOUTP(1).GT.0)
     .CALL        OUTMP      (EPSTN,IITER,MTOTG,MTOTV,MVFIX,NELEM,
     .                        NGAUS,NOFIX,NOUTP,NPOIN,NVFIX,STRSG,
     .                        TDISP,TREAC)
C
C*** IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS
C
      IF(NCHEK.EQ.0) GO TO 100
   90 CONTINUE
C
C***
C
      IF(NALGO.EQ.2) GO TO 100
```

```
      STOP
  100 CALL        OUTMP      (EPSTN,IITER,MTOTG,MTOTV,MVFIX,NELEM,
     .                        NGAUS,NOFIX,NOUTP,NPOIN,NVFIX,STRSG,
     .                        TDISP,TREAC)
   70 CONTINUE
   20 CONTINUE
   10 CONTINUE
      STOP
      END
```

9.5.3 Subroutine CONVMP

This routine establishes whether a solution has converged with reference to some displacement or residual force norm.

```
      SUBROUTINE CONVMP     (ASDIS,ELOAD,IITER,IFDIS,IFRES,LNODS,
     .                       MELEM,MEVAB,MTOTV,NCHEK,NCDIS,NCRES,
     .                       NDOFN,NELEM,NEVAB,NNODE,NPOIN,NTOTV,
     .                       REFOR,TOFOR,TDISP,TLOAD,TOLER)
C***********************************************************************
C
C*** ESTABLISHES WHETHER A SOLUTION HAS CONVERGED WITH
C*** REFERENCE TO SOME DISPLACEMENT OR RESIDUAL FORCE NORM
C
C***********************************************************************
      DIMENSION ADIDF(3),ASDIS(MTOTV),ELOAD(MELEM,MEVAB),LNODS(MELEM,9),
     .          NCDIS(4),NCRES(4),REFDF(3),REFOR(MTOTV),TDIDF(3),
     .          TDISP(MTOTV),TLOAD(MELEM,MEVAB),TOFDF(3),TOFOR(MTOTV)
      WRITE(6,606) IITER
  606 FORMAT(///,' IN CONVER',10X,'ITERATION NUMBER',I3,/)
C*** COMPUTE ELEMENT RESIDUAL FORCES
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=TLOAD(IELEM,IEVAB)-ELOAD(IELEM,IEVAB)
C*** SET CONVERGENCE CODE TO ZERO
      NCHEK=0
C*** DISPLACEMENT CONVERGENCE CHECK
      IF(IFDIS.EQ.0) GOTO 1000
C*** COMPUTE TOTAL AND DIRECTIONAL NORMS OF DISPLACEMENTS
      ADITO=0.0
      TDITO=0.0
      CALL        VZERO      (NDOFN,ADIDF)
      CALL        VZERO      (NDOFN,TDIDF)
      NPOSI=0
      DO 20 IPOIN=1,NPOIN
      DO 20 IDOFN=1,NDOFN
      NPOSI=NPOSI+1
      ADIDF(IDOFN)=ADIDF(IDOFN)+ASDIS(NPOSI)*ASDIS(NPOSI)
   20 TDIDF(IDOFN)=TDIDF(IDOFN)+TDISP(NPOSI)*TDISP(NPOSI)
      DO 30 IDOFN=1,NDOFN
      ADITO=ADITO+ADIDF(IDOFN)
      TDITO=TDITO+TDIDF(IDOFN)
      ADIDF(IDOFN)=SQRT(ADIDF(IDOFN))
   30 TDIDF(IDOFN)=SQRT(TDIDF(IDOFN))
      ADITO=SQRT(ADITO)
      TDITO=SQRT(TDITO)
C*** CHECK FOR CONVERGENCE AND PRINT ERRORS PER CENT
      DO 40 IDOFN=1,NDOFN
      IF(TDIDF(IDOFN).EQ.0.0) GOTO 40
```

```
      TDIDF(IDOFN)=100.*ADIDF(IDOFN)/TDIDF(IDOFN)
      IF(NCDIS(IDOFN).NE.0.AND.TDIDF(IDOFN).GT.TOLER) NCHEK=1
      IF(NCDIS(IDOFN).EQ.0) TDIDF(IDOFN)=-TDIDF(IDOFN)
   40 CONTINUE
      IF(TDITO.EQ.0.0) GOTO 50
      TDITO=100.*ADITO/TDITO
      IF(NCDIS(4).NE.0.AND.TDITO.GT.TOLER) NCHEK=1
      IF(NCDIS(4).EQ.0) TDITO=-TDITO
   50 CONTINUE
      WRITE(6,600)
      WRITE(6,601) (TDIDF(IDOFN),IDOFN=1,NDOFN)
  600 FORMAT(1X,'DISPLACEMENT CHANGE NORM',//)
  601 FORMAT(1X,5(E10.3,5X))
      WRITE(6,602)
  602 FORMAT(5X,'TOTAL')
      WRITE(6,603) TDITO
  603 FORMAT(3X,E10.3)
C*** RESIDUAL CONVERGENCE CHECK
 1000 IF(IFRES.EQ.0) GOTO 2000
C*** ASSEMBLE TOTAL AND RESIDUAL FORCE VECTORS
      DO 1 ITOTV=1,NTOTV
      REFOR(ITOTV)=0.0
    1 TOFOR(ITOTV)=0.0
      DO 60 IELEM=1,NELEM
      KEVAB=0
      DO 60 INODE=1,NNODE
      LOCNO=IABS(LNODS(IELEM,INODE))
      DO 60 IDOFN=1,NDOFN
      KEVAB=KEVAB+1
      NPOSI=(LOCNO-1)*NDOFN+IDOFN
      TOFOR(NPOSI)=TOFOR(NPOSI)+TLOAD(IELEM,KEVAB)
   60 REFOR(NPOSI)=REFOR(NPOSI)+ELOAD(IELEM,KEVAB)
C*** COMPUTE TOTAL AND DIRECTIONAL NORMS OF RESIDUAL AND TOTAL FORCE
      REFTO=0.0
      TOFTO=0.0
      CALL        VZERO      (NDOFN,REFDF)
      CALL        VZERO      (NDOFN,TOFDF)
      NPOSI=0
      DO 70 IPOIN=1,NPOIN
      DO 70 IDOFN=1,NDOFN
      NPOSI=NPOSI+1
      REFDF(IDOFN)=REFDF(IDOFN)+REFOR(NPOSI)*REFOR(NPOSI)
   70 TOFDF(IDOFN)=TOFDF(IDOFN)+TOFOR(NPOSI)*TOFOR(NPOSI)
      DO 80 IDOFN=1,NDOFN
      REFTO=REFTO+REFDF(IDOFN)
      TOFTO=TOFTO+TOFDF(IDOFN)
      REFDF(IDOFN)=SQRT(REFDF(IDOFN))
   80 TOFDF(IDOFN)=SQRT(TOFDF(IDOFN))
      REFTO=SQRT(REFTO)
      TOFTO=SQRT(TOFTO)
C*** CHECK FOR CONVERGENCE AND PRINT ERRORS PER CENT
      DO 90 IDOFN=1,NDOFN
      IF(TOFDF(IDOFN).EQ.0.0) GOTO 90
      TOFDF(IDOFN)=100.*REFDF(IDOFN)/TOFDF(IDOFN)
      IF(NCRES(IDOFN).NE.0.AND.TOFDF(IDOFN).GT.TOLER) NCHEK=1
      IF(NCRES(IDOFN).EQ.0) TOFDF(IDOFN)=-TOFDF(IDOFN)
   90 CONTINUE
      IF(TOFTO.EQ.0.0) GOTO 100
      TOFTO=100.*REFTO/TOFTO
      IF(NCRES(4).NE.0.AND.TOFTO.GT.TOLER) NCHEK=1
      IF(NCRES(4).EQ.0) TOFTO=-TOFTO
  100 CONTINUE
      WRITE(6,604)
      WRITE(6,601) (TOFDF(IDOFN),IDOFN=1,NDOFN)
```

```
      WRITE(6,602)
      WRITE(6,603) TOFTO
  604 FORMAT(1X,'RESIDUAL NORM',//)
C*** PRINT CONVERGENCE CODE
 2000 WRITE(6,605) NCHEK
  605 FORMAT(1X,'CONVERGENCE CODE',I4,//)
      RETURN
      END
```

9.5.4 Subroutine DIMMP

This subroutine sets up the dimensions which must agree with the size of the arrays in subroutine FEMP.

```
      SUBROUTINE DIMMP      (MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,
     .                       MSTIF,MTOTG,MTOTV,MVFIX,NDIME,NDOFN,
     .                       NPROP,NSTRE)
C**********************************************************************
C
C***  SETS UP DYNAMIC DIMENSIONS - MUST AGREE WITH DIMENSIONS
C***  IN FEMP
C
C**********************************************************************
      MBUFA = 10
      MELEM = 25
      MFRON = 40
      MMATS = 10
      MPOIN = 80
      MSTIF=(MFRON*MFRON-MFRON)/2.0+MFRON
      MTOTG = MELEM*9
      NDOFN = 3
      MTOTV = MPOIN*NDOFN
      MVFIX = 40
      NDIME=2
      NPROP = 8
      NSTRE = 5
      MEVAB = NDOFN*9
      RETURN
      END
```

9.5.5 Subroutine FLOWMP

This subroutine determines the yield function derivatives $[\partial F/\partial M_x, \partial F/\partial M_y, \partial F/\partial M_{xy}]^T$ for nonlayered Mindlin plates of Von Mises or Tresca material. This routine is almost identical to the corresponding one given in Chapter 7 for plane stress, plane strain and axisymmetric problems.

```
      SUBROUTINE FLOWMP     (ABETA,AVECT,DEVIA,DMATX,DVECT,HARDS,
     .                       NCRIT,SINT3,STEFF,THETA,VARJ2)
C**********************************************************************
C
C***  DETERMINES YIELD FUNCTION DERIVATIVES  FOR MINDLIN PLATES
C***  1 VON MISES
C***  2 TRESCA
C
C**********************************************************************
```

```
C
      DIMENSION AVECT(5),DEVIA(4),DMATX(3,3),DVECT(5),
     .          VECA1(3),VECA2(3),VECA3(3)
C
C*** DETERMINE THE VECTOR DERIVATIVE OF F FOR VON-MISES
      SINTH=SIN(THETA)
      COSTH=COS(THETA)
      ROOT3=1.73205080757
C
C*** CALCULATE VECTOR A1
C
      VECA1(1)=0.333333333333
      VECA1(2)=0.333333333333
      VECA1(3)=0.0
C
C*** CALCULATE VECTOR A2
C
      DO 10 ISTRE=1,3
   10 VECA2(ISTRE)=DEVIA(ISTRE)/(2.0*STEFF)
      VECA2(3)=DEVIA(3)/STEFF
C
C*** CALCULATE VECTOR A3
C
      VECA3(1)=DEVIA(2)*DEVIA(4)+VARJ2/3.0
      VECA3(2)=DEVIA(1)*DEVIA(4)+VARJ2/3.0
      VECA3(3)=-2.0*DEVIA(3)*DEVIA(4)
      GO TO (1,2) NCRIT
C
C*** VON MISES
C
    1 CONS1=0.0
      CONS2=ROOT3
      CONS3=0.0
      GO TO 40
C
C*** TRESCA
C
    2 CONS1=0.0
      ABTHE=ABS(THETA*57.29577951308)
      IF(ABTHE.LT.29.0) GO TO 20
      CONS2=ROOT3
      CONS3=0.0
      GO TO 40
   20 CONS2=2.0*(COSTH+SINTH*SINT3/SQRT(1.0-SINT3*SINT3))
      CONS3=ROOT3*SINTH/(VARJ2*SQRT(1.0-SINT3*SINT3))
   40 CONTINUE
      DO 50 ISTRE=1,3
   50 AVECT(ISTRE)=CONS1*VECA1(ISTRE)+CONS2*VECA2(ISTRE)+CONS3*
     .VECA3(ISTRE)
C
C*** DETERMINE THE  VECTOR D
C
      DENOM=HARDS
      DO 120 ISTRE=1,3
      DVECT(ISTRE)=0.0
      DO 110 JSTRE=1,3
  110 DVECT(ISTRE)=DVECT(ISTRE)+DMATX(ISTRE,JSTRE)*AVECT(JSTRE)
  120 DENOM=DENOM+AVECT(ISTRE)*DVECT(ISTRE)
      ABETA=1.0/DENOM
      RETURN
      END
```

9.5.6 Subroutine GRADMP

This subroutine evaluates displacement gradients $\partial w/\partial x$, $\partial w/\partial y$, $\partial\theta_x/\partial x$, $\partial\theta_x/\partial y$, $\partial\theta_y/\partial x$ and $\partial\theta_y/\partial y$.

```
      SUBROUTINE GRADMP    (CARTD,DGRAD,ELDIS,NDOFN,NNODE)
C***********************************************************************
C
C*** FORM TOTAL DISPLACEMENTS GRADIENTS
C
C***********************************************************************
      DIMENSION CARTD(2,9),DGRAD(6),ELDIS(3,9)
C*** ZERO DGRAD
      CALL VZERO(6,DGRAD)
C*** FORM TOTAL DISPLACEMENTS GRADIENTS
      DO 10 INODE=1,NNODE
      DNIDX=CARTD(1,INODE)
      DNIDY=CARTD(2,INODE)
      DO 10 IDOFN=1,NDOFN
      IPOSN=NDOFN+IDOFN
      CONST=ELDIS(IDOFN,INODE)
      DGRAD(IDOFN)=DGRAD(IDOFN)+DNIDX*CONST
   10 DGRAD(IPOSN)=DGRAD(IPOSN)+DNIDY*CONST
      RETURN
      END
```

9.5.7 Subroutine INVMP

This subroutine evaluates the Mindlin plate bending moment invariants. It also evaluates the effective moment for the Tresca and Von Mises materials.

```
      SUBROUTINE INVMP     (DEVIA,NCRIT,SINT3,STEFF,STEMP,THETA,
     .                      VARJ2,YIELD)
C***********************************************************************
C
C***  CALCULATE MINDLIN PLATE STRESS RESULTANT INVARIANTS
C
C***********************************************************************
      DIMENSION STEMP(5),DEVIA(4)
      SMEAN=(STEMP(1)+STEMP(2))/3.0
      DEVIA(1)=STEMP(1)-SMEAN
      DEVIA(2)=STEMP(2)-SMEAN
      DEVIA(3)=STEMP(3)
      DEVIA(4)=-SMEAN
      VARJ2=DEVIA(3)*DEVIA(3)+0.5*(DEVIA(1)*DEVIA(1)+DEVIA(2)*DEVIA(2)
     . +DEVIA(4)*DEVIA(4))
      VARJ3=DEVIA(4)*(DEVIA(4)*DEVIA(4)-VARJ2)
      STEFF=SQRT(VARJ2)
      SINT3=-2.5980762113*VARJ3/(VARJ2*STEFF)
      THETA=ASIN(SINT3)/3.0
      GO TO (1,2) NCRIT
C*** VON MISES
    1 YIELD=1.73205080757*STEFF
      RETURN
C*** TRESCA
    2 YIELD=2.0*COS(THETA)*STEFF
      RETURN
      END
```

9.5.8 Subroutine MINDPB

This subroutine simply reads some additional information required for controlling the convergence check and inserting additional constraints for the Heterosis element.

```
      SUBROUTINE MINDPB     (IFDIS,IFFIX,IFRES,LNODS,MELEM,MTOTV,
     .                       NCDIS,NCRES,NELEM,NTYPE)
C**********************************************************************
C
C***  READS ADDITIONAL DATA FOR MINDLIN PLATE ANALYSIS
C
C**********************************************************************
      DIMENSION DERIV(2,9),IFFIX(MTOTV),
     .          LNODS(MELEM,9),NCDIS(4),NCRES(4),SHAPE(9)
C
C*** READ DATA CONTROLLING CONVERGENCE CHECK
C
   10 READ(5,900) IFDIS,(NCDIS(I),I=1,4)
     .           ,IFRES,(NCRES(I),I=1,4)
  900 FORMAT(5I1)
      WRITE(6,901) IFDIS,(NCDIS(I),I=1,4)
     .            ,IFRES,(NCRES(I),I=1,4)
  901 FORMAT(/,23H CONVERGENCE PARAMETERS,/,
     .       8H IFDIS =,I2,5X,8H NCDIS =,4I1,/,
     .       8H IFRES =,I2,5X,8H NCRES =,4I1,//)
C*** INSERT ADDITIONAL CONSTRAINT FOR HETEROSIS ELEMENT
      IF(NTYPE.NE.5) GO TO 30
      DO 20 IELEM=1,NELEM
      LNODE=LNODS(IELEM,9)
      NLOCA=LNODE*3-2
   20 IFFIX(NLOCA)=-1
   30 CONTINUE
      RETURN
      END
```

9.5.9 Subroutine NODEXY

This subroutine evaluates midside nodes for straight sided 8 and 9–node quadrilateral elements. In the original subroutine described in Section 6.4.1 this routine also evaluated the coordinates of the central node. Here, as we are choosing a hierarchical formulation, the values at the central node and the departures from the interpolated Serendipity values are always taken as zero.

Thus the revised subroutine NODEXY is almost identical to its namesake given earlier in Section 6.4.1 and is listed below.

```
      SUBROUTINE NODEXY     (COORD,LNODS,MELEM,MPOIN,NDIME,NELEM,
     .                       NNODE)
C**********************************************************************
C
C***  INTERPOLATES MIDSIDE NODE COORDINATES FOR 8-NODED ELEMENTS
C***  INTERPOLATES CENTRAL AND MIDSIDE NODE COORDINATES FOR
C***  9-NODE ELEMENTS PROVIDED THAT THE SIDES ARE STRAIGHT
C
C**********************************************************************
```

```
      DIMENSION COORD(MPOIN,2),LNODS(MELEM,9)
      IF(NNODE.EQ.4) GO TO 60
C
C*** LOOP OVER EACH ELEMENT
C
      DO 30 IELEM=1,NELEM
C
C*** LOOP OVER EACH ELEMENT EDGE
C
      NNOD1=NNODE
      IF(NNODE.EQ.8) NNOD1=9
      DO 20 INODE=1,NNOD1,2
      IF(INODE.EQ.9.AND.NNODE.EQ.8) GO TO 30
      IF(INODE.EQ.9) GO TO 50
C
C*** COMPUTE THE NODE NUMBER OF THE FIRST NODE
C
      NODST=LNODS(IELEM,INODE)
      IGASH=INODE+2
      IF(IGASH.GE.NNODE) IGASH=1
C
C*** COMPUTE THE NODE NUMBER OF THE LAST NODE
C
      NODFN=LNODS(IELEM,IGASH)
      MIDPT=INODE+1
C
C*** COMPUTE THE NODE NUMBER OF THE INTERMEDIATE NODE
C
      NODMD=LNODS(IELEM,MIDPT)
      TOTAL=ABS(COORD(NODMD,1))+ABS(COORD(NODMD,2))
C
C*** IF THE COORDINATES OF THE INTERMEDIATE NODE ARE BOTH ZERO
C    INTERPOLATE BY A STRAIGHT LINE
C
      IF(TOTAL.GT.0.0) GO TO 20
      KOUNT=1
   10 COORD(NODMD,KOUNT)=(COORD(NODST,KOUNT)+COORD(NODFN,KOUNT))/2.0
      KOUNT=KOUNT+1
      IF(KOUNT.EQ.2) GO TO 10
   20 CONTINUE
   50 LNODE=LNODS(IELEM,INODE)
   30 CONTINUE
   60 CONTINUE
      RETURN
      END
```

9.5.10 Subroutine OUTMP

This subroutine outputs nodal displacements and reactions and also the Gauss point stress resultants.

```
      SUBROUTINE OUTMP    (EPSTN,IITER,MTOTG,MTOTV,MVFIX,NELEM,
     .                     NGAUS,NOFIX,NOUTP,NPOIN,NVFIX,STRSG,
     .                     TDISP,TREAC)
C***********************************************************************
C
C***  OUTPUT DISPLACEMENTS,REACTIONS AND GAUSS POINT STRESS
C***  RESULTANTS FOR EP MINDLIN PLATE ANALYSIS
C
C***********************************************************************
```

```
      DIMENSION EPSTN(MTOTG),GPCOD(2,9),NOFIX(MVFIX),NOUTP(2),
     .          STRSG(5,MTOTG),TDISP(MTOTV),TREAC(MVFIX,3)
      KOUTP=NOUTP(1)
      IF(IITER.GT.1) KOUTP=NOUTP(2)
C
C*** OUTPUT DISPLACEMENTS
C
      IF(KOUTP.LT.1) GO TO 10
      WRITE(6,900)
  900 FORMAT(1H0,5X,13HDISPLACEMENTS)
      WRITE(6,950)
  950 FORMAT(1H0,6X,4HNODE,6X,5HDISP.,8X,7HXZ-ROT.,7X,7HYZ-ROT.)
      DO 20 IPOIN=1,NPOIN
      NGASH=IPOIN*3
      NGISH=NGASH-3+1
   20 WRITE(6,910) IPOIN,(TDISP(IGASH),IGASH=NGISH,NGASH)
  910 FORMAT(I10,3E14.6)
   10 CONTINUE
C
C*** OUTPUT REACTIONS
C
      IF(KOUTP.LT.2) GO TO 30
      WRITE(6,920)
  920 FORMAT(1H0,5X,9HREACTIONS)
      WRITE(6,960)
  960 FORMAT(1H0,6X,4HNODE,6X,5HFORCE,3X,9HXZ-MOMENT,5X,9HYZ-MOMENT)
      DO 40 IVFIX=1,NVFIX
   40 WRITE(6,910) NOFIX(IVFIX),(TREAC(IVFIX,IDOFN),IDOFN=1,3)
   30 CONTINUE
C
C*** OUTPUT STRESSES
C
      IF(KOUTP.LT.3) GO TO 50
      REWIND 3
      WRITE(6,970)
  970 FORMAT(1H0,5X,8HSTRESSES)
      WRITE(6,980)
  980 FORMAT(1H0,4HG.P.,2X,8HX-COORD.,2X,8HY-COORD.,3X,8HX-MOMENT,4X,
     .8HY-MOMENT,3X,9HXY-MOMENT,3X,
     .13HEFF.PL.STRAIN)
      KGAUS=0
      DO 60 IELEM=1,NELEM
      READ(3)GPCOD
      KELGS=0
      WRITE(6,930)IELEM
  930 FORMAT(1H0,5X,13HELEMENT NO. =,I5)
      DO 60 IGAUS=1,NGAUS
      DO 60 JGAUS=1,NGAUS
      KGAUS=KGAUS+1
      KELGS=KELGS+1
      WRITE(6,940)KELGS,(GPCOD(IDIME,KELGS),IDIME=1,2),
     .(STRSG(ISTRE,KGAUS),ISTRE=1,3),EPSTN(KGAUS)
  940 FORMAT(I5,2F10.4,6E12.5)
   60 CONTINUE
   50 CONTINUE
      RETURN
      END
```

9.5.11 Subroutine RESMP

This subroutine evaluates the residual nodal forces. The structure of this routine is similar to that given in Chapter 7 for the other two dimensional elasto-plastic applications and it is illustrated in Fig. 9.3.

```
      SUBROUTINE RESMP     (ASDIS,COORD,EFFST,ELOAD,EPSTN,LNODS,
     .                      MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,
     .                      NCRIT,NELEM,NEVAB,NGAUS,NNODE,PROPS,
     .                      STRSG)
C***********************************************************************
C
C*** EVALUATES EQUIVALENT NODAL FORCES FOR THE STRESS RESULTANTS
C*** IN MINDLIN PLATES DURING EP ANALYSIS
C
C***********************************************************************
      DIMENSION ASDIS(MTOTV),AVECT(5),CARTD(2,9),
     .          COORD(MPOIN,2),DERIV(2,9),DESIG(5),DEVIA(4),
     .          DVECT(5),
     .          EFFST(MTOTG),ELCOD(2,9),
     .          ELDIS(3,9),ELOAD(MELEM,27),EPSTN(MTOTG),GPCOD(2,9),
     .          LNODS(MELEM,9),MATNO(MELEM),POSGP(4),
     .          PROPS(MMATS,8),SGTOT(5),SHAPE(9),SIGMA(5),
     .          STRES(5),STRSG(5,MTOTG),WEIGP(4),
     .          DFLEX(3,3),DSHER(2,2),BFLEI(3,3),BSHEI(2,3),
     .          DUMMY(3,3),FORCE(3),DGRAD(6)
      NTIME=1
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=0.0
      KGAUS=0
      LGAUS=0
      DO 20 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 190 INODE =1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      NPOSN=(LNODE-1)*3
      DO 30 IDOFN=1,3
      NPOSN=NPOSN+1
   30 ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
      DO 180 IDIME=1,2
  180 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
  190 CONTINUE
      KGASP=0
      CALL      MODPB      (DFLEX,DUMMY,DSHER,LPROP,MMATS,PROPS,
     .                           0,     1,     1)
      CALL GAUSSQ      (NGAUS,POSGP,WEIGP)
      DO 40 IGAUS=1,NGAUS
      DO 40 JGAUS=1,NGAUS
      BRING=1.0
      KGAUS=KGAUS+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      CALL       SFR2       (DERIV,ETASP,EXISP,NNODE,SHAPE)
      KGASP=KGASP+1
      CALL       JACOB2     (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                       KGASP,NNODE,SHAPE)
```

```
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      CALL      GRADMP    (CARTD,DGRAD,ELDIS,    3,NNODE)
      CALL      STRMP     (CARTD,DFLEX,DGRAD,DSHER,ELDIS,NNODE,
     .                     SHAPE,STRES,     1,     0)
      PREYS=PROPS(LPROP,6)+EPSTN(KGAUS)*PROPS(LPROP,7)
      DO 150 ISTRE=1,3
      DESIG(ISTRE)=STRES(ISTRE)
  150 SIGMA(ISTRE)=STRSG(ISTRE,KGAUS)+STRES(ISTRE)
      CALL      INVMP      (DEVIA,NCRIT,SINT3,STEFF,SIGMA,THETA,
     .                      VARJ2,YIELD)
      ESPRE=EFFST(KGAUS)-PREYS
      IF(ESPRE.GE.0.0) GO TO 50
      ESCUR=YIELD-PREYS
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=ESCUR/(YIELD-EFFST(KGAUS))
      GO TO 70
   50 ESCUR=YIELD-EFFST(KGAUS)
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=1.0
   70 MSTEP=ESCUR*8.0/PROPS(LPROP,6)+1.0
      ASTEP=MSTEP
      REDUC=1.0-RFACT
      DO 80 ISTRE=1,3
      SGTOT(ISTRE)=STRSG(ISTRE,KGAUS)+REDUC*STRES(ISTRE)
   80 STRES(ISTRE)=RFACT*STRES(ISTRE)/ASTEP
      DO 90 ISTEP=1,MSTEP
      CALL      INVMP      (DEVIA,NCRIT,SINT3,STEFF,SGTOT,THETA,
     .                      VARJ2,YIELD)
      HARDS=PROPS(LPROP,7)
      CALL      FLOWMP     (ABETA,AVECT,DEVIA,DFLEX,DVECT,HARDS,
     .                      NCRIT,SINT3,STEFF,THETA,VARJ2)
      AGASH=0.0
      DO 100 ISTRE=1,3
  100 AGASH=AGASH+AVECT(ISTRE)*STRES(ISTRE)
      DLAMD=AGASH*ABETA
      IF(DLAMD.LT.0.0) DLAMD=0.0
      BGASH=0.0
      DO 110 ISTRE=1,3
      BGASH=BGASH+AVECT(ISTRE)*SGTOT(ISTRE)
  110 SGTOT(ISTRE)=SGTOT(ISTRE)+STRES(ISTRE)-DLAMD*DVECT(ISTRE)
   90 EPSTN(KGAUS)=EPSTN(KGAUS)+DLAMD*BGASH/YIELD
      DO 120 ISTRE=1,3
  120 DESIG(ISTRE)=SGTOT(ISTRE)-STRSG(ISTRE,KGAUS)
      CALL      INVMP      (DEVIA,NCRIT,SINT3,STEFF,SGTOT,THETA,
     .                      VARJ2,YIELD)
      CURYS=PROPS(LPROP,6)+EPSTN(KGAUS)*PROPS(LPROP,7)
      IF(YIELD.GT.CURYS) BRING=CURYS/YIELD
   60 DO 130 ISTRE=1,3
      SGTOT(ISTRE)=BRING*(STRSG(ISTRE,KGAUS)+DESIG(ISTRE))
  130 STRSG(ISTRE,KGAUS)=SGTOT(ISTRE)
      EFFST(KGAUS)=BRING*YIELD
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE
C    ELEMENT NODES
      DO 140 INODE=1,NNODE
C*** ZERO FORCE VECTOR
      CALL      VZERO      (3,FORCE)
      CALL      BMATPB    (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                        0,     1,     0)
      FORCE(2)=(BFLEI(1,2)*SGTOT(1)+BFLEI(3,2)*SGTOT(3))*DAREA
     .         +FORCE(2)
      FORCE(3)=(BFLEI(2,3)*SGTOT(2)+BFLEI(3,3)*SGTOT(3))*DAREA
     .         +FORCE(3)
      IPOSN=(INODE-1)*3+1
```

```
      DO 135 IDOFN=2,3
      IPOSN=IPOSN+1
  135 ELOAD(IELEM,IPOSN)=ELOAD(IELEM,IPOSN)+FORCE(IDOFN)
  140 CONTINUE
   40 CONTINUE
C
C*** CALCULATE FORCES ASSOCIATED WITH SHEAR DEFORMATION
C
      NGAUM=NGAUS-1
      CALL GAUSSQ      (NGAUM,POSGP,WEIGP)
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      KGASP=0
      DO 300 IGAUS=1,NGAUM
      DO 300 JGAUS=1,NGAUM
      LGAUS=LGAUS+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      CALL       SFR2      (DERIV,ETASP,EXISP,NNODE,SHAPE)
      KGASP=KGASP+1
      CALL       JACOB2    (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                      KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      CALL      GRADMP    (CARTD,DGRAD,ELDIS,    3,NNODE)
      CALL      STRMP     (CARTD,DFLEX,DGRAD,DSHER,ELDIS,NNODE,
     .                     SHAPE,STRES,    0,     1)
      DO 310 ISTRE=4,5
      SGTOT(ISTRE)=STRSG(ISTRE,LGAUS)+STRES(ISTRE)
  310 STRSG(ISTRE,LGAUS)=SGTOT(ISTRE)
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES
C
      DO 320 INODE=1,NNODE
C*** ZERO FORCE VECTOR
      CALL VZERO(3,FORCE)
      CALL      BMATPB    (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                         0,     0,     1)
      FORCE(1)=(BSHEI(1,1)*SGTOT(4)+BSHEI(2,1)*SGTOT(5))*DAREA
     .         +FORCE(1)
      FORCE(2)=(BSHEI(1,2)*SGTOT(4))*DAREA+FORCE(2)
      FORCE(3)=(BSHEI(2,3)*SGTOT(5))*DAREA+FORCE(3)
      IPOSN=(INODE-1)*3
      DO 315 IDOFN=1,3
      IPOSN=IPOSN+1
  315 ELOAD(IELEM,IPOSN)=ELOAD(IELEM,IPOSN)+FORCE(IDOFN)
  320 CONTINUE
  300 CONTINUE
   20 CONTINUE
      RETURN
      END
```

9.5.12 Subroutine SFR2

This subroutine evaluates the shape functions and their derivatives for 4, 8 and 9-node quadrilateral isoparametric elements. The 9-node element is treated as a hierarchical element as described in Section 9.3.2. This enables the Heterosis element to be easily incorporated.

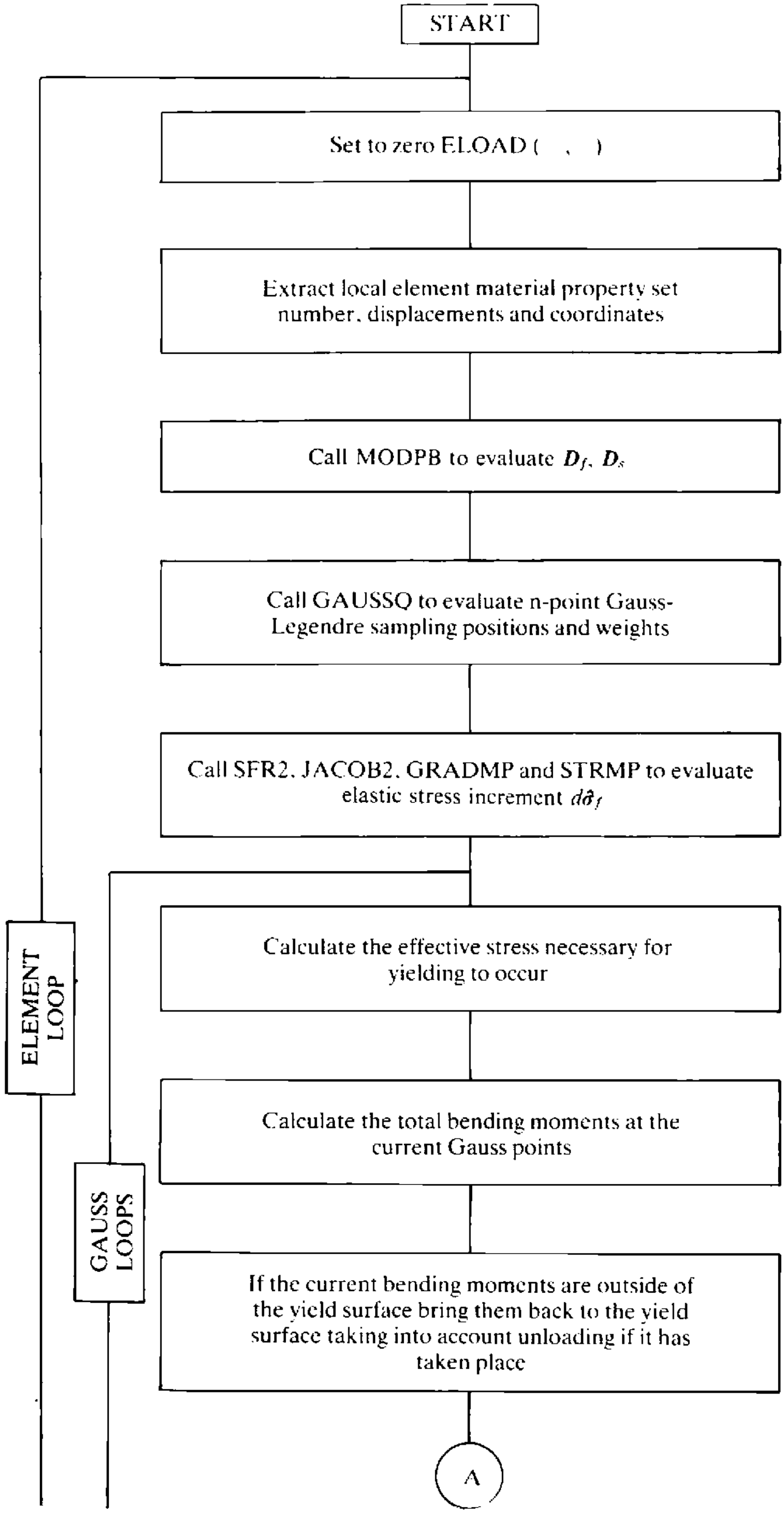


Fig. 9.3 Overall structure of subroutine RESMP.

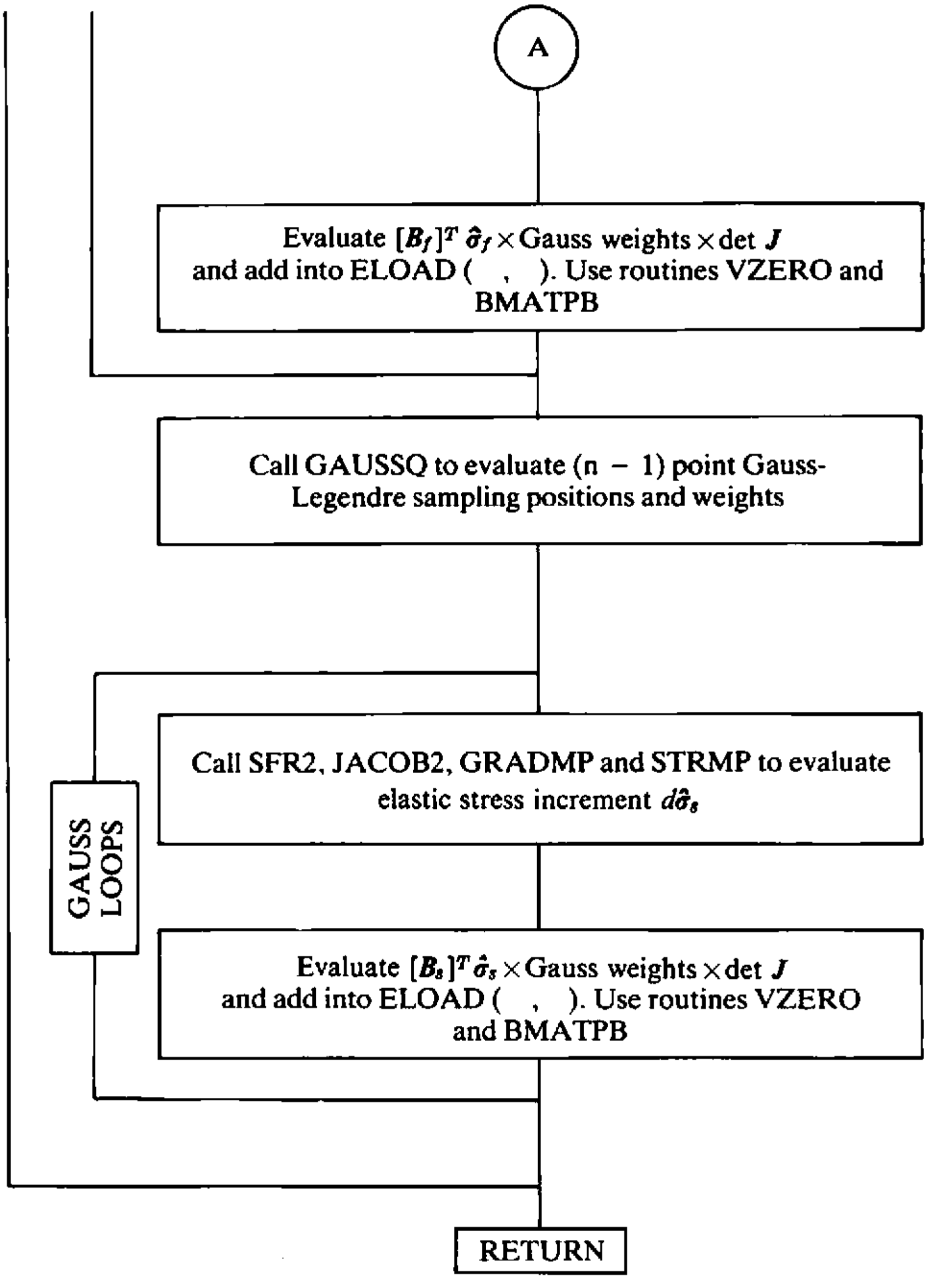


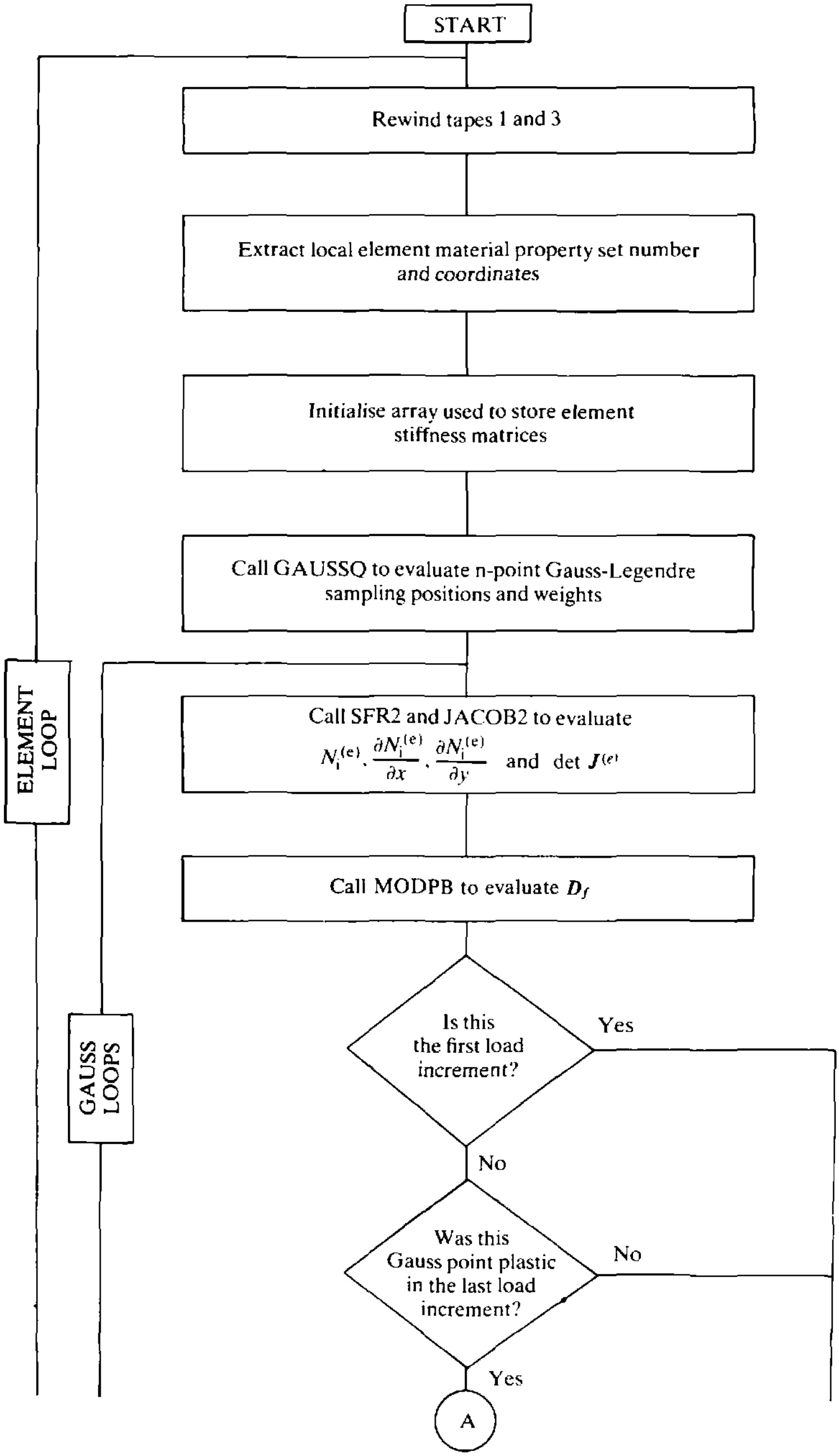
Fig. 9.3 Overall structure of subroutine RESMP (continued).

Subroutine SFR2 is identical to its namesake given earlier in Section 6.4.3 except that SFR2 72–118 are replaced by SFRH 67–73.

```
      IF(NNODE.EQ.8) RETURN                                          SFR2  67
C*** BUBBLE FUNCTION FOR HIERARCHICAL AND HETEROSIS ELEMENTS          SFRH  68
      SHAPE(9)=(1.0-SS)*(1.0-TT)                                     SFRH  69
      DERIV(1,9)=-S2*(1.0-TT)                                        SFRH  70
      DERIV(2,9)=-T2*(1.0-SS)                                        SFRH  71
      RETURN                                                         SFRH  72
      END                                                            SFRH  73
```

9.5.13 Subroutine STIFMP

This routine evaluates the stiffness matrix for the nonlayered elasto-plastic Mindlin plate elements. The overall structure is shown in Fig. 9.4.


START
Rewind tapes 1 and 3
Extract local element material property set number and coordinates
Initialise array used to store element stiffness matrices
Call GAUSSQ to evaluate n-point Gauss-Legendre sampling positions and weights
ELEMENT LOOP
Call SFR2 and JACOB2 to evaluate
$N_i^{(e)}, \dfrac{\partial N_i^{(e)}}{\partial x}, \dfrac{\partial N_i^{(e)}}{\partial y}$ and det $\boldsymbol{J}^{(e)}$
Call MODPB to evaluate $\boldsymbol{D}_f$
GAUSS LOOPS
Is this the first load increment?
Yes
No
Was this Gauss point plastic in the last load increment?
No
Yes
A

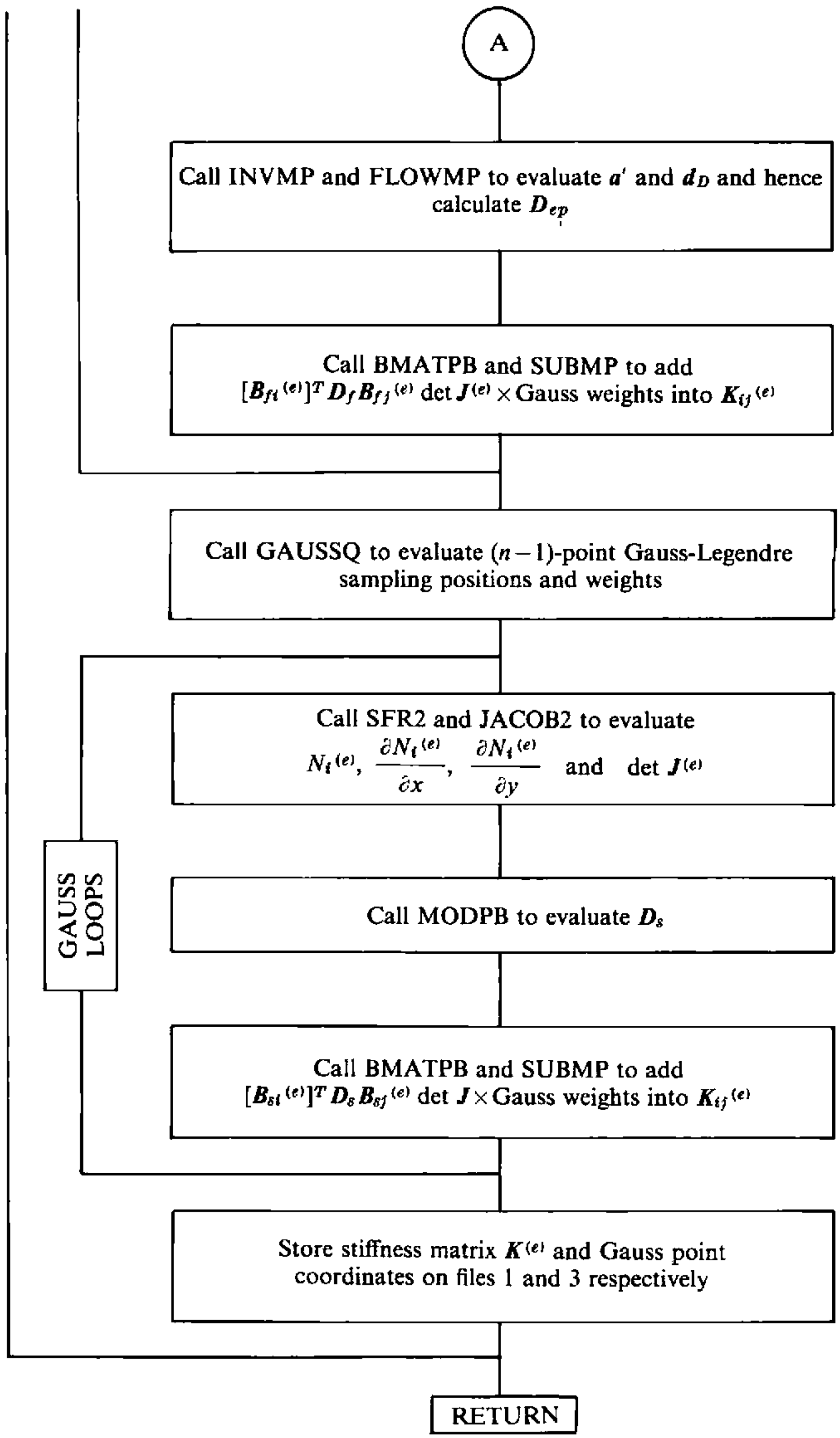


Fig. 9.4 Overall structure of subroutine STIFMP (continued).

```
      SUBROUTINE STIFMP    (COORD,EPSTN,IINCS,LNODS,MATNO,MELEM,
     .                      MEVAB,MMATS,MPOIN,MTOTG,NCRIT,NELEM,
     .                      NEVAB,NGAUS,NNODE,PROPS,STRSG)
C**********************************************************************
C
C***  EVALUATE STIFFNESS MATRICES FOR NON-LAYERED
C***  ELASTO-PLASTIC MINDLIN PLATE ELEMENTS
C
C**********************************************************************
      DIMENSION AVECT(5),
     .          CARTD(2,9),COORD(MPOIN,2),
     .          DERIV(2,9),DEVIA(4),DVECT(5),ELCOD(2,9),
     .          EPSTN(MTOTG),ESTIF(27,27),GPCOD(2,9),LNODS(MELEM,9),
     .          MATNO(MELEM),POSGP(4),PROPS(MMATS,8),SHAPE(9),STRES(5),
     .          STRSG(5,MTOTG),WEIGP(4),
     .          DFLEX(3,3),DSHER(2,2),BFLEI(3,3),BFLEJ(3,3),
     .          BSHEI(2,3),BSHEJ(2,3),DUMMY(3,3)
      REWIND 1
      REWIND 3
      KGAUS=0
C
C*** LOOP OVER EACH ELEMENT
C
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 10 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      LNODE=IABS(LNODE)
      DO 10 IDIME=1,2
   10 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
C
      DO 20 IEVAB=1,NEVAB
      DO 20 JEVAB=1,NEVAB
   20 ESTIF(IEVAB,JEVAB)=0.0
C
C*** EVALUATE PART OF STIFFNESS MATRIX
C    ASSOCIATED WITH BENDING DEFORMATION
C
      KGASP=0
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL      GAUSSQ     (NGAUS,POSGP,WEIGP)

      DO 50 IGAUS=1,NGAUS
      DO 50 JGAUS=1,NGAUS
      KGASP=KGASP+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL AREA,ETC
C
      CALL      SFR2       (DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL      JACOB2     (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                      KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
```

```
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL      MODPB      (DFLEX,DUMMY,DSHER,LPROP,MMATS,PROPS,
     .                          0,     1,     0)
      IF(IINCS.EQ.1) GO TO 80
      KGAUS=KGAUS+1
      IF(EPSTN(KGAUS).EQ.0.0) GO TO 80
      DO 90 ISTRE=1,3
   90 STRES(ISTRE)=STRSG(ISTRE,KGAUS)
      HARDS=PROPS(LPROP,7)
      CALL      INVMP      (DEVIA,NCRIT,SINT3,STEFF,STRES,THETA,
     .                      VARJ2,YIELD)
      CALL      FLOWMP     (ABETA,AVECT,DEVIA,DFLEX,DVECT,HARDS,
     .                      NCRIT,SINT3,STEFF,THETA,VARJ2)
      DO 100 ISTRE=1,3
      DO 100 JSTRE=1,3
  100 DFLEX(ISTRE,JSTRE)=DFLEX(ISTRE,JSTRE)-ABETA*DVECT(ISTRE)*
     . DVECT(JSTRE)
   80 CONTINUE
C
C*** CALCULATE THE ELEMENT STIFFNESSES
C
      DO 30 INODE=1,NNODE
      CALL      BMATPB     (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                          0,     1,     0)
      DO 30 JNODE=INODE,NNODE
      CALL      BMATPB     (BFLEJ,DUMMY,BSHEJ,CARTD,JNODE,SHAPE,
     .                          0,     1,     0)
   30 CALL      SUBMP      (BFLEI,BFLEJ,DAREA,DFLEX,ESTIF,INODE,
     .                      JNODE,     3,     3,     3)
   50 CONTINUE
C
C*** EVALUATE PART OF STIFFNESS MATRIX
C    ASSOCIATED WITH SHEAR DEFORMATION
C
      KGASP=0
      NGAUM=NGAUS-1
C
C*** ENTER LOOPS FOR AREA INTEGRATION
C
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL      GAUSSQ      (NGAUM,POSGP,WEIGP)
      DO 51 IGAUS=1,NGAUM
      DO 51 JGAUS=1,NGAUM
      KGASP=KGASP+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL AREA,ETC
C
      CALL      SFR2       (DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL      JACOB2     (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                      KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL      MODPB      (DFLEX,DUMMY,DSHER,LPROP,MMATS,PROPS,
     .                          0,     0,     1)
C
C*** EVALUATE ELEMENT STIFFNESSES
```

```
C
      DO 31 INODE=1,NNODE
      CALL      BMATPB     (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                             0,    0,     1)
      DO 31 JNODE=INODE,NNODE
      CALL     BMATPB      (BFLEJ,DUMMY,BSHEJ,CARTD,JNODE,SHAPE,
     .                             0,    0,     1)
   31 CALL      SUBMP      (BSHEI,BSHEJ,DAREA,DSHER,ESTIF,INODE,
     .                      JNODE,    3,     2,     3)
   51 CONTINUE
C
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX
C
      DO 60 IEVAB=1,NEVAB
      DO 60 JEVAB=IEVAB,NEVAB
   60 ESTIF(JEVAB,IEVAB)=ESTIF(IEVAB,JEVAB)
C
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX AND SAMPLING POINT
C    COORDINATES FOR EACH ELEMENT ON DISC FILE
C
C
      WRITE(1) ESTIF
      WRITE(3) GPCOD
   70 CONTINUE
      RETURN
      END
```

9.5.14 Subroutine STRMP

This subroutine evaluates the bending moments and shear forces for Mindlin plates.

```
      SUBROUTINE STRMP     (CARTD,DFLEX,DGRAD,DSHER,ELDIS,NNODE,
     .                      SHAPE,STRES,IFFLE,IFSHE)
C*********************************************************************
C
C*** EVALUATES STRESS RESULTANTS FOR MINDLIN PLATE
C
C*********************************************************************
      DIMENSION CARTD(2,9),DFLEX(3,3),DGRAD(6),DSHER(2,2),
     .          ELDIS(3,9),SHAPE(9),STRES(5)
C*** ZERO STRESS VECTOR
      CALL      VZERO     (5,STRES)
C*** EVALUATE ROTATIONS AT GAUSS POINT , IF NEEDED
      IF(IFSHE.EQ.0) GOTO 50
      XZROT=0.0
      YZROT=0.0
      DO 30 INODE=1,NNODE
      XZROT=XZROT+SHAPE(INODE)*ELDIS(2,INODE)
   30 YZROT=YZROT+SHAPE(INODE)*ELDIS(3,INODE)
C*** EVALUATE BENDING STRESS RESULTANTS
   50 IF(IFFLE.EQ.0) GOTO 60
      EFLXX=-DGRAD(2)
      EFLYY=-DGRAD(6)
      EFLXY=-(DGRAD(3)+DGRAD(5))
      STRES(1)=DFLEX(1,1)*EFLXX+DFLEX(1,2)*EFLYY
      STRES(2)=DFLEX(2,1)*EFLXX+DFLEX(2,2)*EFLYY
      STRES(3)=DFLEX(3,3)*EFLXY
```

```
C*** EVALUATE SHEAR STRESS RESULTANTS
   60 IF(IFSHE.EQ.0) RETURN
      ESHXX=DGRAD(1)-XZROT
      ESHYY=DGRAD(4)-YZROT
      STRES(4)=DSHER(1,1)*ESHXX
      STRES(5)=DSHER(2,2)*ESHYY
      RETURN
      END
```

9.5.15 Subroutine SUBMP

This subroutine evaluates $[B_i]^T D\,[B_j]\,det\,J \times$ Gauss weights and is used in the evaluation of the element stiffness matrices.

```
      SUBROUTINE SUBMP      (BIMAT,BJMAT,DAREA,DMATX,ESTIF,INODE,
     .                       JNODE,NCOLI,NROIJ,NCOLJ)
C***********************************************************************
C
C*** CARRY OUT MATRIX MULTIPLICATION
C
C***********************************************************************
      DIMENSION BIMAT(NROIJ,NCOLI),BJMAT(NROIJ,NCOLJ),
     .          DMATX(NROIJ,NROIJ),DBMAT(3,3),
     .          ESTIF(27,27),SBSTF(3,3)
C*** EVALUATE  D*BJ
      DO 10 J=1,NCOLJ
      DO 10 I=1,NROIJ
      DBMAT(I,J)=0.0
      DO 10 K=1,NROIJ
   10 DBMAT(I,J)=DBMAT(I,J)+DMATX(I,K)*BJMAT(K,J)
C*** EVALUATE  BIT*(D*BJ)
      DO 20 J=1,NCOLJ
      DO 20 I=1,NCOLI
      SBSTF(I,J)=0.0
      DO 20 K=1,NROIJ
   20 SBSTF(I,J)=SBSTF(I,J)+BIMAT(K,I)*DBMAT(K,J)
C*** ASSEMBLE SBSTF INTO ELEMENT STIFFNESS MATRIX
      IFROW=0
      JFCOL=0
      IFROW=(INODE-1)*3+IFROW
      JFCOL=(JNODE-1)*3+JFCOL
      DO 30 I=1,NCOLI
      IRSUB=IFROW+I
      DO 30 J=1,NCOLJ
      JCSUB=JFCOL+J
   30 ESTIF(IRSUB,JCSUB)=ESTIF(IRSUB,JCSUB)+SBSTF(I,J)*DAREA
      RETURN
      END
```

9.5.16 Subroutines VZERO and ZEROMP

These routines simply set to zero the components of various vectors and arrays.

```
      SUBROUTINE VZERO      (NCOMP,VECTO)
C***********************************************************************
C
C***  ZEROES VECTOR VECTO
C
C***********************************************************************
      DIMENSION VECTO(NCOMP)
      DO 10 ICOMP=1,NCOMP
   10 VECTO(ICOMP)=0.0
      RETURN
      END
```

```
      SUBROUTINE ZEROMP     (EFFST,ELOAD,EPSTN,MELEM,MEVAB,MTOTG,
     .                       MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NGAUS,
     .                       NTOTG,NTOTV,NVFIX,STRSG,TDISP,TFACT,
     .                       TLOAD,TREAC)
C**********************************************************************
C
C***  ZERO EFFST,ELOAD,EPSTN,STRSG,TDISP,TFACT,TLOAD,TREAC
C
C**********************************************************************
      DIMENSION ELOAD(MELEM,MEVAB),STRSG(5,MTOTG),TDISP(MTOTV),
     .          TLOAD(MELEM,MEVAB),TREAC(MVFIX,3),EPSTN(MTOTG),
     .          EFFST(MTOTG)
      TFACT=0.0
      DO 30 IELEM=1,NELEM
      DO 30 IEVAB=1,NEVAB
      ELOAD(IELEM,IEVAB)=0.0
   30 TLOAD(IELEM,IEVAB)=0.0
      DO 40 ITOTV=1,NTOTV
   40 TDISP(ITOTV)=0.0
      DO 50 IVFIX=1,NVFIX
      DO 50 IDOFN=1,NDOFN
   50 TREAC(IVFIX,IDOFN)=0.0
      DO 60 ITOTG=1,NTOTG
      EPSTN(ITOTG)=0.0
      EFFST(ITOTG)=0.0
      DO 60 ISTR1=1,5
   60 STRSG(ISTR1,ITOTG)=0.0
      RETURN
      END
```

9.6 Software for the layered approach

9.6.1 Overall program structure

The overall program structure for the elasto-plastic Mindlin plate bending analysis program using the layered approach is given in Fig. 9.5. This program is named MINDLAY.

The program can solve problems of the same size as those solved by program MINDLIN. A maximum of 26 layers is allowed.

All new routines are now documented and these include: FEAM, DEPMPA, LAYMPA, MDMPA, OUTMPA, RESMPA, STIMPA and STRMPA. The outer routines, which have been described earlier, include ALGOR, BMATPB, CHECK1, CHECK2, ECHO, FRONT, INCREM, INPUT, JACOB2 and NODEXY.

The files which are used in the program are 5 (cardreader), 6 (lineprinter) and 1, 2, 3, 4, 8 (scratch files).

9.6.2 Subroutine FEAM

This routine organises the calling of the main routines in sequence.

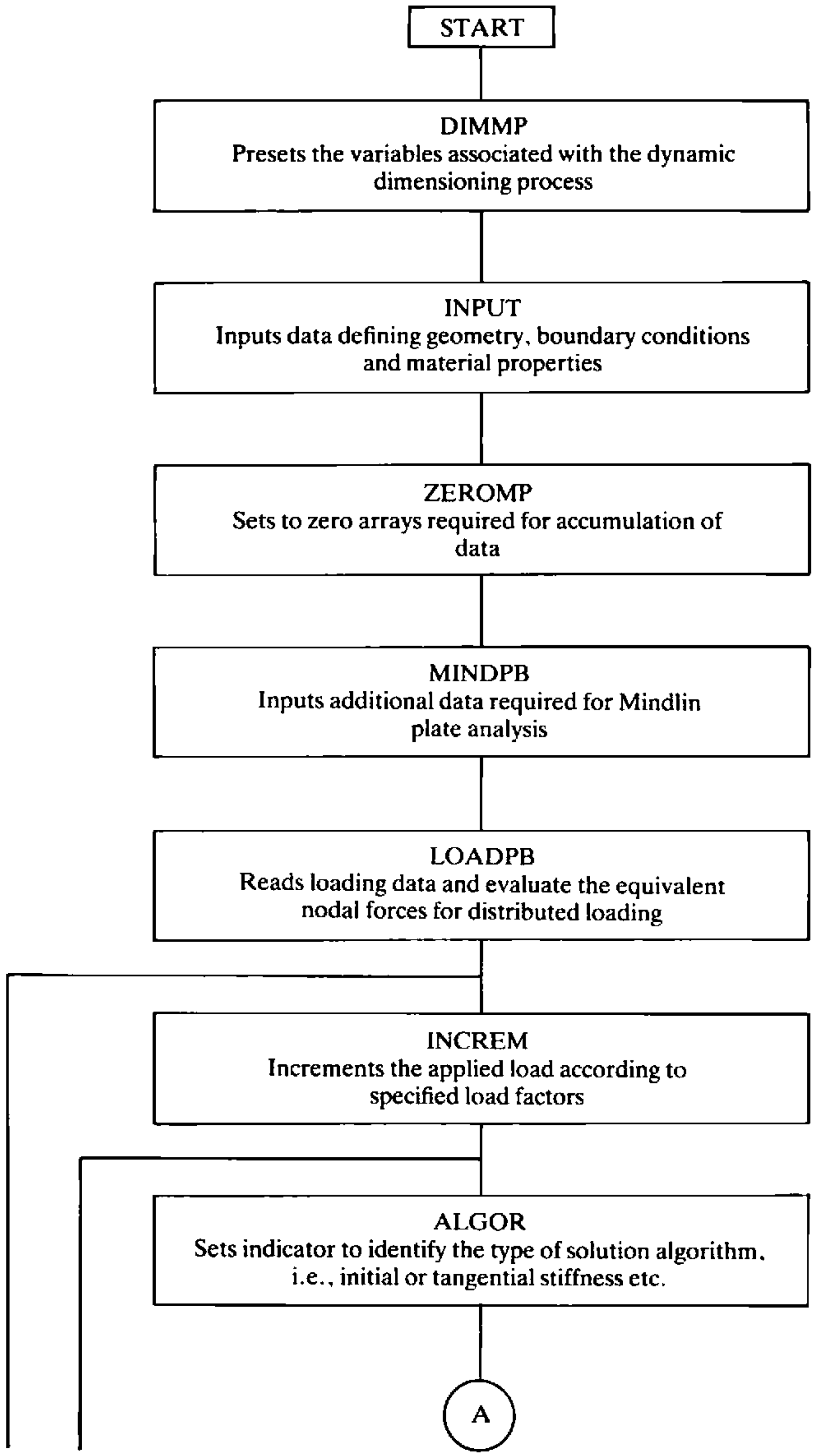


Fig. 9.5 Overall program structure of program MINDLAY.

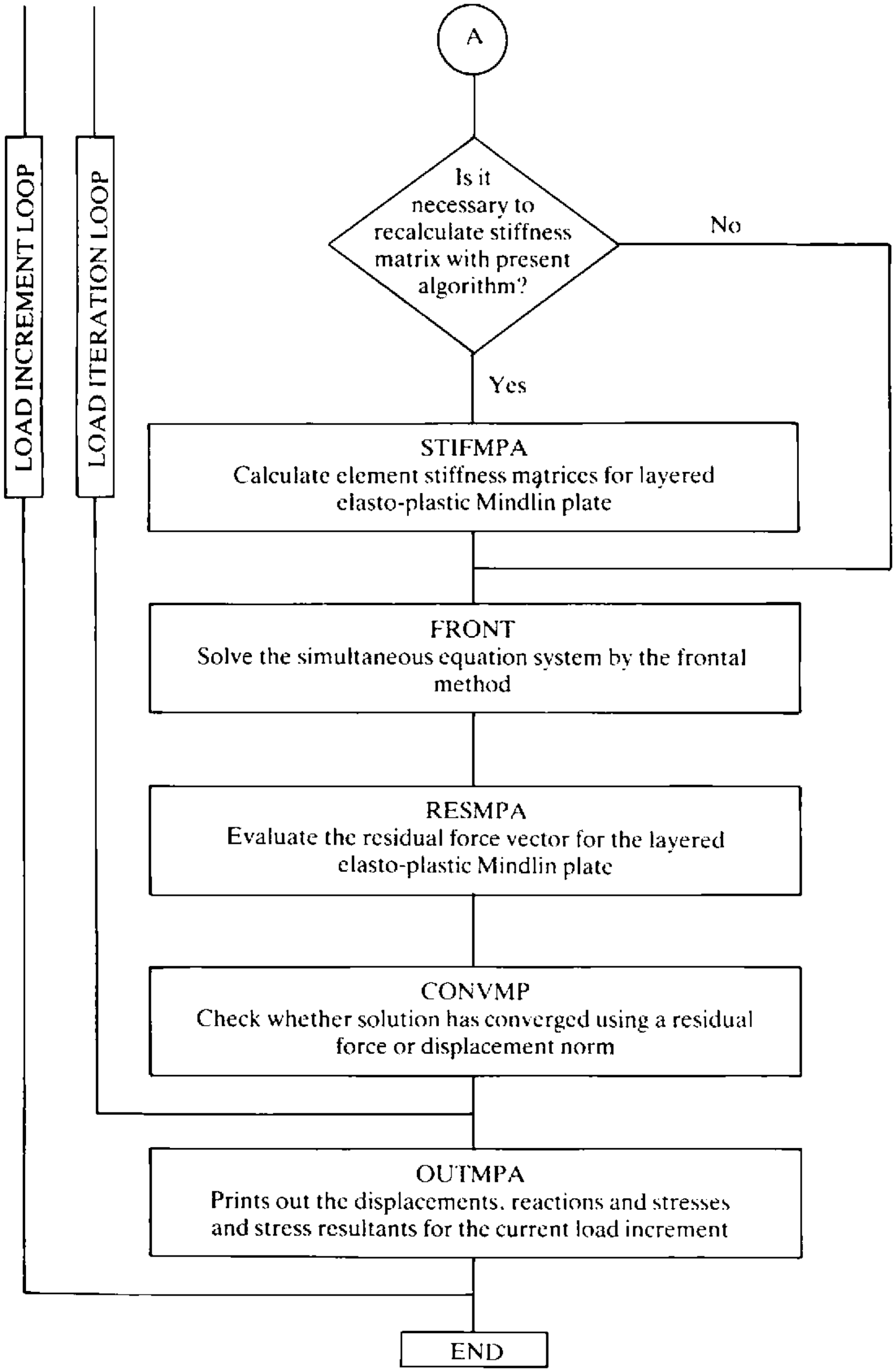


Fig. 9.5 Overall program structure of program MINDLAY (continued).

```
      PROGRAM FEAM(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
     .TAPE1,TAPE2,TAPE3,TAPE4,TAPE8,TAPE9)
C***********************************************************************
C
C***  ELASTO-PLASTIC ANALYSIS OF LAYERED MINDLIN PLATES USING
C***  4-,8-, 9-NODED OR HETEROSIS ISOPARAMETRIC QUADRILATERALS
C
C***********************************************************************
      DIMENSION ASDIS(240),COORD(80,2),EFFST(225),ELOAD(25,27),
     .          EPSTN(225),ESTIF(27,27),
     .          EQRHS(10),EQUAT(40,10),FIXED(240),
     .          IFFIX(240),GLOAD(40),GSTIF(860),LNODS(25,9),LOCEL(27),
     .          MATNO(25),NACVA(40),NAMEV(10),NCDIS(4),NCRES(4),
     .          NDEST(27),NDFRO(25),NOFIX(40),NOUTP(2),NPIVO(10),
     .          POSGP(4),PRESC(40,3),PROPS(10,8),REFOR(240),
     .          RLOAD(25,27),STRSG(5,225),TOFOR(240),
     .          TDISP(240),TLOAD(25,27),TREAC(40,3),VECRV(40),
     .          WEIGP(4)
C
C*** PRESET VARIABLES ASSOCIATED WITH DYNAMIC DIMENSIONS
C
      CALL       DIMMP      (MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,
     .                        MSTIF,MTOTG,MTOTV,MVFIX,NDIME,NDOFN,
     .                        NPROP,NSTRE)
C
C*** CALL THE SUBROUTINE WHICH READS MOST OF THE PROBLEM DATA
C
      CALL       INPUT      (COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,
     .                        MFRON,MMATS,MPOIN,MTOTV,MVFIX,NALGO,
     .                        NCRIT,NDFRO,NDIME,NDOFN,NELEM,NEVAB,
     .                        NGAUS,NLAPS,NINCS,NMATS,NNODE,NOFIX,
     .                        NPOIN,NPROP,NSTRE,NSTR1,NSWIT,NTOTG,
     .                        NTOTV,NTYPE,NVFIX,POSGP,PRESC,PROPS,
     .                        WEIGP)
C
C*** INITIALIZE ARRAYS TO ZERO
C
      CALL       ZEROMP     (EFFST,ELOAD,EPSTN,MELEM,MEVAB,MTOTG,
     .                        MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NGAUS,
     .                        NTOTG,NTOTV,NVFIX,STRSG,TDISP,TFACT,
     .                        TLOAD,TREAC)
C
C***
C
      CALL       MINDPB     (IFDIS,IFFIX,IFRES,LNODS,MELEM,MTOTV,
     .                        NCDIS,NCRES,NELEM,NTYPE)
C
C
C
C*** COMPUTE LOAD AFTER READING RELEVANT EXTRA DATA
C
      CALL       LOADPB     (COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,
     .                        NELEM,NEVAB,NGAUS,NNODE,NPOIN,PROPS,
     .                        RLOAD)
C
C*** LOOP OVER EACH INCREMENT
C
      DO 70 IINCS=1,NINCS
C
C*** READ DATA FOR CURRENT INCREMENT
C
      CALL       INCREM     (ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,
     .                        MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NOUTP,
     .                        NOFIX,NTOTV,NVFIX,PRESC,RLOAD,TFACT,
     .                        TLOAD,TOLER)
```

```
C
C*** LOOP OVER EACH ITERATION
C
      DO 90 IITER=1,MITER
C
C*** CALL ROUTINE WHICH SELECTS SOLUTION ALGORITHM VARIABLE KRESL
C
      CALL        ALGOR      (FIXED,IINCS,IITER,KRESL,MTOTV,NALGO,
     .                        NTOTV)
C
C*** CHECK WHETHER A NEW EVALUATION OF THE STIFFNESS MATRICES IS NEEDED
C
      IF(KRESL.EQ.1)
     .CALL        STIMPA     (COORD,EPSTN,IINCS,LNODS,MATNO,MELEM,
     .                        MEVAB,MMATS,MPOIN,MTOTG,NCRIT,NELEM,
     .                        NEVAB,NGAUS,NNODE,NLAPS,PROPS,STRSG)
C
C*** SOLVE EQUATIONS
C
      CALL        FRONT      (ASDIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,
     .                        IFFIX,IINCS,IITER,GLOAD,GSTIF,KRESL,
     .                        LNODS,LOCEL,MBUFA,MELEM,MEVAB,MFRON,
     .                        MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,
     .                        NDOFN,NELEM,NEVAB,NNODE,NOFIX,NPIVO,
     .                        NPOIN,NTOTV,TDISP,TLOAD,TREAC,VECRV)
C
C*** CALCULATE RESIDUAL FORCES
C
      CALL        RESMPA     (ASDIS,COORD,EFFST,ELOAD,EPSTN,LNODS,
     .                        MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,
     .                        NCRIT,NELEM,NEVAB,NGAUS,NNODE,NLAPS,
     .                        PROPS,STRSG)
C
C*** CHECK FOR CONVERGENCE
C
      CALL        CONVMP     (ASDIS,ELOAD,IITER,IFDIS,IFRES,LNODS,
     .                        MELEM,MEVAB,MTOTV,NCHEK,NCDIS,NCRES,
     .                        NDOFN,NELEM,NEVAB,NNODE,NPOIN,NTOTV,
     .                        REFOR,TOFOR,TDISP,TLOAD,TOLER)
C
C*** OUTPUT RESULTS IF REQUIRED

C
      IF(IITER.EQ.1.AND.NOUTP(1).GT.0)
     .CALL        OUTMPA     (EPSTN,IITER,MTOTG,MTOTV,MVFIX,NELEM,
     .                        NGAUS,NLAPS,NOFIX,NOUTP,NPOIN,NVFIX,
     .                        STRSG,TDISP,TREAC)
C
C*** IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS
C
      IF(NCHEK.EQ.0) GO TO 100
   90 CONTINUE
C
C***
C
      IF(NALGO.EQ.2) GO TO 100
      STOP
  100 CALL        OUTMPA     (EPSTN,IITER,MTOTG,MTOTV,MVFIX,NELEM,
     .                        NGAUS,NLAPS,NOFIX,NOUTP,NPOIN,NVFIX,
     .                        STRSG,TDISP,TREAC)
   70 CONTINUE
   20 CONTINUE
   10 CONTINUE
      STOP
      END
```

9.6.3 Subroutine CHECK1 (revised)

In program MINDLAY we remove card CEK1 25 from subroutine CHECK1 because NLAPS (the number of layers) replaces NSTRE in subroutine INPUT. The variable NSTRE is set in subroutine DIMMP (see Section 9.5.4).

9.6.4 Subroutine DEPMPA

This subroutine sets up the layered discretisation.

```
      SUBROUTINE DEPMPA    (DEPTH,LPROP,MMATS,NLAYR,PROPS)              DEPT   1
C***********************************************************************DEPT   2
C                                                                       DEPT   3
C***  SET UP LAYRED DISCRETIZATION                                      DEPT   4
C                                                                       DEPT   5
C***********************************************************************DEPT   6
       DIMENSION PROPS(MMATS,8),DEPTH(26)                               DEPT   7
C                                                                       DEPT   8
C                                                                       DEPT   9
      NLAY1=NLAYR+1                                                     DEPT  10
      ALAYR=NLAYR                                                       DEPT  11
      THICK=PROPS(LPROP,3)                                              DEPT  12
      CONS1=THICK/ALAYR                                                 DEPT  13
      CONS2=-THICK/2.0                                                  DEPT  14
      KOUNT=0                                                           DEPT  15
      DO 10 ILAYR=1,NLAY1                                               DEPT  16
      DEPTH(ILAYR)=CONS2+CONS1*KOUNT                                    DEPT  17
   10 KOUNT=KOUNT+1                                                     DEPT  18
      RETURN                                                            DEPT  19
      END                                                               DEPT  20
```

9.6.5 Subroutine LAYMPA

This subroutine evaluates $\hat{D}_f$ and $\hat{D}_s$ using the mid-ordinate rule.

```
      SUBROUTINE  LAYMPA   (DEPTH,DFLEF,DSHES,EPSTN,IINCS,KGAUS,        LAYR   1
     .                      LPROP,MMATS,MTOTG,NCRIT,NLAYR,PROPS,        LAYR   2
     .                      STRSG,JFFLE)                                LAYR   3
C***********************************************************************LAYR   4
C                                                                       LAYR   5
C*** CALCULATES THE D-MATRIX INTEGRATED OVER                            LAYR   6
C***  THE DEPTH                                                         LAYR   7
C                                                                       LAYR   8
C***********************************************************************LAYR   9
      DIMENSION AVECT(3),DEPTH(26),DEVIA(4),DFLEF(3,3),                 LAYR  10
     .          DPLAN(3,3),DVECT(3),                                    LAYR  11
     .          DSHER(2,2),DSHES(2,2),EPSTN(MTOTG),PROPS(MMATS,8),      LAYR  12
     .          SGTOT(5),STRSG(5,MTOTG)                                 LAYR  13
C                                                                       LAYR  14
C                                                                       LAYR  15
      IF(JFFLE.EQ.0) GO TO 100                                          LAYR  16
      HARDS=PROPS(LPROP,7)                                              LAYR  17
C                                                                       LAYR  18
C*** ZERO D MATRIX FOR FLEXURE                                          LAYR  19
C                                                                       LAYR  20
      DO 20 ISTRE=1,3                                                   LAYR  21
      DO 20 JSTRE=1,3                                                   LAYR  22
   20 DFLEF(ISTRE,JSTRE)=0.0                                            LAYR  23
C                                                                       LAYR  24
C*** LOOP AROUND LAYERS                                                 LAYR  25
C                                                                       LAYR  26
```

```
      DO 30 ILAYR=1,NLAYR
      KGAUS=KGAUS+1
      JLAYR=ILAYR+1
C
C*** EVALUATE Z-COORDINATES FOR CURRENT LAYER
C
      DEPT1=DEPTH(ILAYR)
      DEPT2=DEPTH(JLAYR)
      CONS3=(DEPT2+DEPT1)*(DEPT2**2-DEPT1**2)/4.0
C
C*** EVALUATE ELASTO-PLASTIC D MATRIX FOR CURRENT LAYER
C
      CALL MDMPA(DPLAN,DSHER,LPROP,MMATS,PROPS,1,0)
      IF(IINCS.EQ.1)GO TO 40
      IF(EPSTN(KGAUS).EQ.0.0)GO TO 40
      DO 50 ISTRE=1,5
   50 SGTOT(ISTRE)=STRSG(ISTRE,KGAUS)
      CALL INVMP(DEVIA,NCRIT,SINT3,STEFF,SGTOT,THETA,VARJ2,YIELD)
      CALL FLOWMP(ABETA,AVECT,DEVIA,DPLAN,DVECT,HARDS,NCRIT,SINT3,
     ,            STEFF,THETA,VARJ2)
      DO 60 ISTRE=1,3
      DO 60 JSTRE=1,3
   60 DPLAN(ISTRE,JSTRE)=DPLAN(ISTRE,JSTRE)-ABETA*DVECT(ISTRE)*
     .DVECT(JSTRE)
   40 CONTINUE
C
C*** SUM D MATRIX OVER ELEMENT DEPTH
C
      DO 70 ISTRE=1,3
      DO 70 JSTRE=1,3
   70 DFLEF(ISTRE,JSTRE)=DFLEF(ISTRE,JSTRE)+CONS3*DPLAN(ISTRE,JSTRE)
   30 CONTINUE
      GO TO 200
C
C*** ZERO D MATRIX FOR SHEAR
C
  100 DO 80 ISTRE=1,2
      DO 80 JSTRE=1,2
   80 DSHES(ISTRE,JSTRE)=0.0
C
C*** EVALUATE ELASTIC D MATRIX
C
      CALL MDMPA(DPLAN,DSHER,LPROP,MMATS,PROPS,0,1)
C
C*** LOOP AROUND LAYERS
C
      DO 90 ILAYR=1,NLAYR
      JLAYR=ILAYR+1
C
C*** EVALUATE Z-COORDINATES FOR CURRENT LAYER
C
      DEPT1=DEPTH(ILAYR)
      DEPT2=DEPTH(JLAYR)
      CONS4=DEPT2-DEPT1
C
C*** SUM D MATRIX OVER ELEMENT DEPTH
C
      DO 110 ISTRE=1,2
      DO 110 JSTRE=1,2
  110 DSHES(ISTRE,JSTRE)=DSHES(ISTRE,JSTRE)+CONS4*DSHER(ISTRE,JSTRE)
   90 CONTINUE
  200 CONTINUE
      RETURN
      END
```

LAYR 10 If JFFLE is zero $\boldsymbol{D}_f'$ is not evaluated. If it is one $\boldsymbol{D}_s'$ is not evaluated.

LAYR 15–17 Initializes $\boldsymbol{D}_f'$.

LAYR 21 Starts the summation loop to form DFLEF, i.e.

$$\hat{\boldsymbol{D}}_f = \sum_{i=1}^{n} \tfrac{1}{4}(z_{i+1}+z_i)(z_{i+1}{}^2-z_i{}^2)\boldsymbol{D}_f'.$$

LAYR 22 Increases the counter for Gauss points in each layer by 1. It is needed to use the effective plastic strain (EPSTN) stresses (STRSG) calculated in RESMPA.

LAYR 27–29 Forms $\frac{1}{4}(z_{i+1}+z_i)(z_{i+1}{}^2-z_i{}^2)$.

LAYR 33–45 Calls MDMPA to get DPLAN and $\boldsymbol{D}_{ep}'$ is formed using INVMP and FLOWMP.

LAYR 49–51 DFLEF is formed.

LAYR 57–59 DSHES is initialised.

LAYR 63 Calls MDMPA to form DSHER.

LAYR 67–74 Starts the summation loop and the integrating constant for DSHES is evaluated, i.e.

$$\hat{\boldsymbol{D}}_s = \sum_{i=1}^{n} (z_{i+1}-z_i)\boldsymbol{D}_s.$$

LAYR 78–81 DSHES is formed.

9.6.6 Subroutine MDMPA

This subroutine evaluates $\boldsymbol{D}_f'$ and $\boldsymbol{D}_s'$.

```
      SUBROUTINE MDMPA      (DPLAN,DSHER,LPROP,MMATS,PROPS,
     .                       IFPLA,IFSHE)
C***********************************************************************
C
C*** CALCULATES MATRIX OF ELASTIC RIGIDITIES FOR EACH LAYER
C*** OF MINDLIN PLATE
C
C***********************************************************************
      DIMENSION DPLAN(3,3),DSHER(2,2),
     .          PROPS(MMATS,8)
      YOUNG=PROPS(LPROP,1)
      POISS=PROPS(LPROP,2)
      THICK=PROPS(LPROP,3)
C*** FORM DPLAN
      IF(IFPLA.EQ.0) GO TO 10
      DO 1 IROWS=1,3
      DO 1 JCOLS=1,3
    1 DPLAN(IROWS,JCOLS)=0.0
      CONST=YOUNG/(1.0-POISS*POISS)
      DPLAN(1,1)=CONST
      DPLAN(2,2)=CONST
      DPLAN(1,2)=CONST*POISS
```

```
      DPLAN(2,1)=CONST*POISS
      DPLAN(3,3)=CONST*(1.0-POISS)/2.0
C*** FORM DSHER
   10 IF(IFSHE.EQ.0) RETURN
      DO 3 IROWS=1,2
      DO 3 JCOLS=1,2
    3 DSHER(IROWS,JCOLS)=0.0
      DSHER(1,1)=YOUNG/(2.4+2.4*POISS)
      DSHER(2,2)=YOUNG/(2.4+2.4*POISS)
      RETURN
      END
```

9.6.7 Subroutine OUTMPA

This subroutine outputs nodal displacements and reactions and also the Gauss point stress resultants and the stresses within each layer. It is very similar to subroutine OUTMP which was described in Section 9.5.7. Statements OUTP 1–3 are replaced by OUTL 1–3 and statements OUTP 56–66 are replaced by statements OUTL 56–67.

```
      SUBROUTINE OUTMPA    (EPSTN,IITER,MTOTG,MTOTV,MVFIX,NELEM,         OUTL   1
     .                      NGAUS,NLAPS,NOFIX,NOUTP,NPOIN,NVFIX,         OUTL   2
     .                      STRSG,TDISP,TREAC)                           OUTL   3
C*********************************************************************OUTL   4
C                                                                      OUTL   5
C***  OUTPUT DISPLACEMENTS,REACTIONS AND GAUSS POINT STRESSES          OUTL   6
C***  IN EACH LAYER FOR EP MINDLIN PLATE ANALYSIS                      OUTL   7
C                                                                      OUTL   8
C*********************************************************************OUTL   9
      DIMENSION EPSTN(MTOTG),GPCOD(2,9),NOFIX(MVFIX),NOUTP(2),         OUTL  10
     .          STRSG(5,MTOTG),TDISP(MTOTV),TREAC(MVFIX,3)             OUTL  11
      KOUTP=NOUTP(1)                                                   OUTL  12
      IF(IITER.GT.1) KOUTP=NOUTP(2)                                    OUTL  13
C                                                                      OUTL  14
C*** OUTPUT DISPLACEMENTS                                              OUTL  15
C                                                                      OUTL  16
      IF(KOUTP.LT.1) GO TO 10                                          OUTL  17
      WRITE(6,900)                                                     OUTL  18
  900 FORMAT(1H0,5X,13HDISPLACEMENTS)                                  OUTL  19
      WRITE(6,950)                                                     OUTL  20
  950 FORMAT(1H0,6X,4HNODE,6X,5HDISP.,8X,7HXZ-ROT.,7X,7HYZ-ROT.)       OUTL  21
      DO 20 IPOIN=1,NPOIN                                              OUTL  22
      NGASH=IPOIN*3                                                    OUTL  23
      NGISH=NGASH-3+1                                                  OUTL  24
   20 WRITE(6,910) IPOIN,(TDISP(IGASH),IGASH=NGISH,NGASH)              OUTL  25
  910 FORMAT(I10,3E14.6)                                               OUTL  26
   10 CONTINUE                                                         OUTL  27
C                                                                      OUTL  28
C*** OUTPUT REACTIONS                                                  OUTL  29
C                                                                      OUTL  30
      IF(KOUTP.LT.2) GO TO 30                                          OUTL  31
      WRITE(6,920)                                                     OUTL  32
  920 FORMAT(1H0,5X,9HREACTIONS)                                       OUTL  33
      WRITE(6,960)                                                     OUTL  34
  960 FORMAT(1H0,6X,4HNODE,6X,5HFORCE,3X,9HXZ-MOMENT,5X,9HYZ-MOMENT)   OUTL  35
      DO 40 IVFIX=1,NVFIX                                              OUTL  36
   40 WRITE(6,910) NOFIX(IVFIX),(TREAC(IVFIX,IDOFN),IDOFN=1,3)         OUTL  37
   30 CONTINUE                                                         OUTL  38
C                                                                      OUTL  39
C*** OUTPUT STRESSES                                                   OUTL  40
```

```
C
      IF(KOUTP.LT.3) GO TO 50
      REWIND 3
      WRITE(6,970)
  970 FORMAT(1H0,5X,8HSTRESSES)
      WRITE(6,980)
  980 FORMAT(1H0,4HG.P.,2X,8HX-COORD.,2X,8HY-COORD.,3X,8HX-MOMENT,4X,
     .8HY-MOMENT,3X,9HXY-MOMENT,3X,
     .13HEFF.PL.STRAIN)
      KGAUS=0
      DO 60 IELEM=1,NELEM
      READ(3)GPCOD
      KELGS=0
      WRITE(6,930)IELEM
  930 FORMAT(1H0,5X,13HELEMENT NO. =,I5)
      DO 60 IGAUS=1,NGAUS
      DO 60 JGAUS=1,NGAUS
      KELGS=KELGS+1
      DO 60 ILAYR=1,NLAPS
      KGAUS=KGAUS+1
      WRITE(6,940)KELGS,(GPCOD(IDIME,KELGS),IDIME=1,2),
     .(STRSG(ISTRE,KGAUS),ISTRE=1,3),EPSTN(KGAUS)
  940 FORMAT(I5,2F10.4,6E12.5)
   60 CONTINUE
   50 CONTINUE
      RETURN
      END
```

9.6.8 Subroutine RESMPA

This routine evaluates the residual forces for the layered Mindlin plate. It is very similar to RESMP described in Section 9.5.10.

```
      SUBROUTINE RESMPA    (ASDIS,COORD,EFFST,ELOAD,EPSTN,LNODS,
     .                      MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,
     .                      NCRIT,NELEM,NEVAB,NGAUS,NNODE,NLAPS,
     .                      PROPS,STRSG)
C***********************************************************************
C
C***  EVALUATES EQUIVALENT NODAL FORCES FOR THE STRESSES
C***  IN LAYERED MINDLIN PLATES DURING EP ANALYSIS
C
C***********************************************************************
      DIMENSION ASDIS(MTOTV),AVECT(5),CARTD(2,9),
     .          COORD(MPOIN,2),DERIV(2,9),DESIG(5),DEVIA(4),
     .          DEPTH(26),DVECT(5),
     .          EFFST(MTOTG),ELCOD(2,9),
     .          ELDIS(3,9),ELOAD(MELEM,27),EPSTN(MTOTG),GPCOD(2,9),
     .          LNODS(MELEM,9),MATNO(MELEM),POSGP(4),
     .          PROPS(MMATS,8),SGTOT(5),SHAPE(9),SIGMA(5),
     .          STRES(5),STRSG(5,MTOTG),TOSPB(5),WEIGP(4),
     .          DPLAN(3,3),DSHER(2,2),BFLEI(3,3),BSHEI(2,3),
     .          DUMMY(3,3),FORCE(3),DGRAD(6)
      NTIME=1
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 ELOAD(IELEM,IEVAB)=0.0
      KGAUS=0
      LGAUS=0
      DO 20 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
```

```
C
C*** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 190 INODE =1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      NPOSN=(LNODE-1)*3
      DO 30 IDOFN=1,3
      NPOSN=NPOSN+1
   30 ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
      DO 180 IDIME=1,2
  180 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
  190 CONTINUE
      KGASP=0
      CALL DEPMPA(DEPTH,LPROP,MMATS,NLAPS,PROPS)
      CALL      MDMPA     (DPLAN,DSHER,LPROP,MMATS,PROPS,
     .                         1,     1)
      CALL GAUSSQ      (NGAUS,POSGP,WEIGP)
      DO 40 IGAUS=1,NGAUS
      DO 40 JGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      CALL       SFR2      (DERIV,ETASP,EXISP,NNODE,SHAPE)
      KGASP=KGASP+1
      CALL       JACOB2    (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                      KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      DO 400 ISTRE=1,3
  400 TOSPB(ISTRE)=0.0
      DO 410 ILAYR=1,NLAPS
      BRING=1.0
      KGAUS=KGAUS+1
      JLAYR=ILAYR+1
      DEPT1=DEPTH(ILAYR)
      DEPT2=DEPTH(JLAYR)
      CONST=0.5*(DEPT2+DEPT1)
      CALL      GRADMP    (CARTD,DGRAD,ELDIS,    3,NNODE)
      CALL      STRMPA    (CARTD,CONST,DPLAN,DGRAD,DSHER,ELDIS,
     .                     NNODE,SHAPE,STRES,     1,     0)
      PREYS=PROPS(LPROP,6)+EPSTN(KGAUS)*PROPS(LPROP,7)
      DO 150 ISTRE=1,3
      DESIG(ISTRE)=STRES(ISTRE)
  150 SIGMA(ISTRE)=STRSG(ISTRE,KGAUS)+STRES(ISTRE)
      CALL       INVMP     (DEVIA,NCRIT,SINT3,STEFF,SIGMA,THETA,
     .                      VARJ2,YIELD)
      ESPRE=EFFST(KGAUS)-PREYS
      IF(ESPRE.GE.0.0) GO TO 50
      ESCUR=YIELD-PREYS
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=ESCUR/(YIELD-EFFST(KGAUS))
      GO TO 70
   50 ESCUR=YIELD-EFFST(KGAUS)
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=1.0
   70 MSTEP=ESCUR*8.0/PROPS(LPROP,6)+1.0
      ASTEP=MSTEP
      REDUC=1.0-RFACT
      DO 80 ISTRE=1,3
      SGTOT(ISTRE)=STRSG(ISTRE,KGAUS)+REDUC*STRES(ISTRE)
   80 STRES(ISTRE)=RFACT*STRES(ISTRE)/ASTEP
      DO 90 ISTEP=1,MSTEP
      CALL       INVMP     (DEVIA,NCRIT,SINT3,STEFF,SGTOT,THETA,
     .                      VARJ2,YIELD)
      HARDS=PROPS(LPROP,7)
      CALL       FLOWMP    (ABETA,AVECT,DEVIA,DPLAN,DVECT,HARDS,
```

```
     .                   NCRIT,SINT3,STEFF,THETA,VARJ2)
      AGASH=0.0
      DO 100 ISTRE=1,3
  100 AGASH=AGASH+AVECT(ISTRE)*STRES(ISTRE)
      DLAMD=AGASH*ABETA
      IF(DLAMD.LT.0.0) DLAMD=0.0
      BGASH=0.0
      DO 110 ISTRE=1,3
      BGASH=BGASH+AVECT(ISTRE)*SGTOT(ISTRE)
  110 SGTOT(ISTRE)=SGTOT(ISTRE)+STRES(ISTRE)-DLAMD*DVECT(ISTRE)
   90 EPSTN(KGAUS)=EPSTN(KGAUS)+DLAMD*BGASH/YIELD
      DO 120 ISTRE=1,3
  120 DESIG(ISTRE)=SGTOT(ISTRE)-STRSG(ISTRE,KGAUS)
      CALL      INVMP     (DEVIA,NCRIT,SINT3,STEFF,SGTOT,THETA,
     .                     VARJ2,YIELD)
      CURYS=PROPS(LPROP,6)+EPSTN(KGAUS)*PROPS(LPROP,7)
      IF(YIELD.GT.CURYS) BRING=CURYS/YIELD
   60 DO 130 ISTRE=1,3
      SGTOT(ISTRE)=BRING*(STRSG(ISTRE,KGAUS)+DESIG(ISTRE))
  130 STRSG(ISTRE,KGAUS)=SGTOT(ISTRE)
      EFFST(KGAUS)=BRING*YIELD
      CONSA=(DEPT2**2-DEPT1**2)/2.0
      DO 440 ISTRE=1,3
  440 TOSPB(ISTRE)=TOSPB(ISTRE)+SGTOT(ISTRE)*CONSA
  410 CONTINUE
      DO 430 ISTRE=1,3
  430 SGTOT(ISTRE)=TOSPB(ISTRE)
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE
C    ELEMENT NODES
      DO 140 INODE=1,NNODE
C*** ZERO FORCE VECTOR
      CALL     VZERO     (3,FORCE)
      CALL     BMATPB    (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                         0,     1,    0)
      FORCE(2)=(BFLEI(1,2)*SGTOT(1)+BFLEI(3,2)*SGTOT(3))*DAREA
     .         +FORCE(2)
      FORCE(3)=(BFLEI(2,3)*SGTOT(2)+BFLEI(3,3)*SGTOT(3))*DAREA
     .         +FORCE(3)
      IPOSN=(INODE-1)*3+1
      DO 135 IDOFN=2,3
      IPOSN=IPOSN+1
  135 ELOAD(IELEM,IPOSN)=ELOAD(IELEM,IPOSN)+FORCE(IDOFN)
  140 CONTINUE
   40 CONTINUE
C
C*** CALCULATE FORCES ASSOCIATED WITH SHEAR DEFORMATION
C
      NGAUM=NGAUS-1
      CALL GAUSSQ     (NGAUM,POSGP,WEIGP)
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      KGASP=0
      DO 300 IGAUS=1,NGAUM
      DO 300 JGAUS=1,NGAUM
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      CALL      SFR2      (DERIV,ETASP,EXISP,NNODE,SHAPE)
      KGASP=KGASP+1
      CALL      JACOB2    (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                     KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      DO 610 ISTRE=4,5
  610 TOSPB(ISTRE)=0.0
```

```
C
C*** LOOP AROUND LAYRS
C
      DO 600 ILAYR=1,NLAPS
      LGAUS=LGAUS+1
      JLAYR=ILAYR+1
      DEPT1=DEPTH(ILAYR)
      DEPT2=DEPTH(JLAYR)
      CONST=1.0
      CALL      GRADMP    (CARTD,DGRAD,ELDIS,     3,NNODE)
      CALL      STRMPA    (CARTD,CONST,DPLAN,DGRAD,DSHER,ELDIS,
     .                     NNODE,SHAPE,STRES,     0,     1)
      DO 310 ISTRE=4,5
      SGTOT(ISTRE)=STRSG(ISTRE,LGAUS)+STRES(ISTRE)
  310 STRSG(ISTRE,LGAUS)=SGTOT(ISTRE)
      CONSB=DEPT2-DEPT1
      DO 620 ISTRE=4,5
  620 TOSPB(ISTRE)=TOSPB(ISTRE)+SGTOT(ISTRE)*CONSB
  600 CONTINUE
      DO 605 ISTRE=4,5
  605 SGTOT(ISTRE)=TOSPB(ISTRE)
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES
C
      DO 320 INODE=1,NNODE
C*** ZERO FORCE VECTOR
      CALL VZERO(3,FORCE)
      CALL      BMATPB    (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                           0,     0,     1)
      FORCE(1)=(BSHEI(1,1)*SGTOT(4)+BSHEI(2,1)*SGTOT(5))*DAREA
     .         +FORCE(1)
      FORCE(2)=(BSHEI(1,2)*SGTOT(4))*DAREA+FORCE(2)
      FORCE(3)=(BSHEI(2,3)*SGTOT(5))*DAREA+FORCE(3)
      IPOSN=(INODE-1)*3
      DO 315 IDOFN=1,3
      IPOSN=IPOSN+1
  315 ELOAD(IELEM,IPOSN)=ELOAD(IELEM,IPOSN)+FORCE(IDOFN)
  320 CONTINUE
  300 CONTINUE
   20 CONTINUE
      RETURN
      END
```

9.6.9 Subroutine STIFMPA

This routine evaluates the stiffness matrices for layered elasto-plastic Mindlin plate elements.

```
      SUBROUTINE STIMPA    (COORD,EPSTN,IINCS,LNODS,MATNO,MELEM,
     .                      MEVAB,MMATS,MPOIN,MTOTG,NCRIT,NELEM,
     .                      NEVAB,NGAUS,NNODE,NLAPS,PROPS,STRSG)
C**********************************************************************
C
C***  EVALUATE STIFFNESS MATRICES FOR LAYREED ELASTO-PLASTIC
C***  MINDLIN PLATE ELEMENTS
C
C**********************************************************************
      DIMENSION CARTD(2,9),COORD(MPOIN,2),
     .          DERIV(2,9),DEPTH(26),ELCOD(2,9),
     .          EPSTN(MTOTG),ESTIF(27,27),GPCOD(2,9),LNODS(MELEM,9),
     .          MATNO(MELEM),POSGP(4),PROPS(MMATS,8),SHAPE(9),
     .          STRSG(5,MTOTG),WEIGP(4),
     .          DFLEX(3,3),DSHER(2,2),BFLEI(3,3),BFLEJ(3,3),
```

```
     .          BSHEI(2,3),BSHEJ(2,3),DUMMY(3,3)
      REWIND 1
      REWIND 3
      KGAUS=0
C
C*** LOOP OVER EACH ELEMENT
C
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 10 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      LNODE=IABS(LNODE)
      DO 10 IDIME=1,2
   10 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
C
      DO 20 IEVAB=1,NEVAB
      DO 20 JEVAB=1,NEVAB
   20 ESTIF(IEVAB,JEVAB)=0.0
      CALL DEPMPA(DEPTH,LPROP,MMATS,NLAPS,PROPS)
C
C*** EVALUATE PART OF STIFFNESS MATRIX
C    ASSOCIATED WITH BENDING DEFORMATION
C
      KGASP=0
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL      GAUSSQ     (NGAUS,POSGP,WEIGP)

      DO 50 IGAUS=1,NGAUS
      DO 50 JGAUS=1,NGAUS
      KGASP=KGASP+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL AREA,ETC
C
      CALL       SFR2       (DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL       JACOB2     (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                       KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL LAYMPA(DEPTH,DFLEX,DSHER,EPSTN,IINCS,KGAUS,LPROP,
     ,           MMATS,MTOTG,NCRIT,NLAPS,PROPS,STRSG,1)
C
C*** CALCULATE THE ELEMENT STIFFNESSES
C
      DO 30 INODE=1,NNODE
      CALL      BMATPB     (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                           0,     1,    0)
      DO 30 JNODE=INODE,NNODE
      CALL      BMATPB     (BFLEJ,DUMMY,BSHEJ,CARTD,JNODE,SHAPE,
     .                           0,     1,    0)
   30 CALL      SUBMP      (BFLEI,BFLEJ,DAREA,DFLEX,ESTIF,INODE,
                            JNODE,    3,    3,    3)
```

```
   50 CONTINUE
C
C*** EVALUATE PART OF STIFFNESS MATRIX
C    ASSOCIATED WITH SHEAR DEFORMATION
C
      KGASP=0
      NGAUM=NGAUS-1
C
C*** ENTER LOOPS FOR AREA INTEGRATION
C
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL      GAUSSQ      (NGAUM,POSGP,WEIGP)
      DO 51 IGAUS=1,NGAUM
      DO 51 JGAUS=1,NGAUM
      KGASP=KGASP+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL AREA,ETC
C
      CALL       SFR2       (DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL       JACOB2     (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                        KGASP,NNODE,SHAPE)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL LAYMPA(DEPTH,DFLEX,DSHER,EPSTN,IINCS,KGAUS,LPROP,
     ,            MMATS,MTOTG,NCRIT,NLAPS,PROPS,STRSG,0)
C
C*** EVALUATE ELEMENT STIFFNESSES
C
      DO 31 INODE=1,NNODE
      CALL       BMATPB     (BFLEI,DUMMY,BSHEI,CARTD,INODE,SHAPE,
     .                           0,     0,     1)
      DO 31 JNODE=INODE,NNODE
      CALL       BMATPB     (BFLEJ,DUMMY,BSHEJ,CARTD,JNODE,SHAPE,
     .                           0,     0,     1)
   31 CALL       SUBMP      (BSHEI,BSHEJ,DAREA,DSHER,ESTIF,INODE,
     .                      JNODE,     3,     2,     3)
   51 CONTINUE
C
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX
C
      DO 60 IEVAB=1,NEVAB
      DO 60 JEVAB=IEVAB,NEVAB
   60 ESTIF(JEVAB,IEVAB)=ESTIF(IEVAB,JEVAB)
C
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX AND SAMPLING POINT
C    COORDINATES FOR EACH ELEMENT ON DISC FILE
C
C
      WRITE(1) ESTIF
      WRITE(3) GPCOD
   70 CONTINUE
      RETURN
      END
```

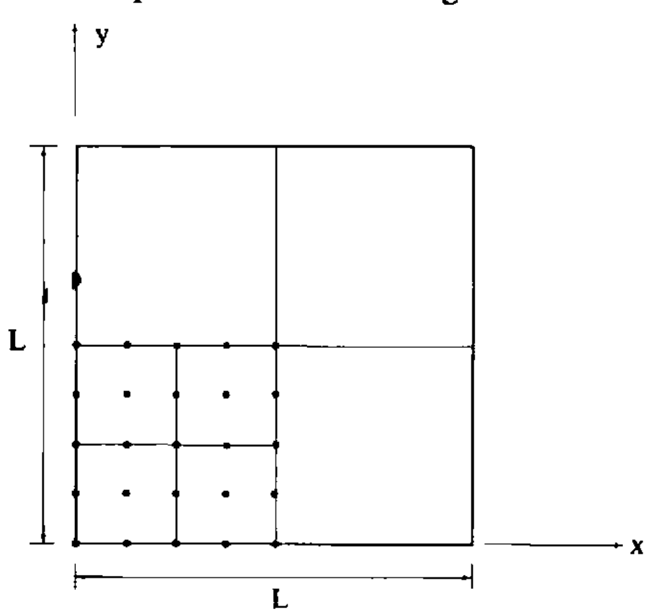
9.6.10 Subroutine STRMPA

This subroutine evaluates the stresses within each layer.

```
      SUBROUTINE STRMPA    (CARTD,CONST,DFLEX,DGRAD,DSHER,ELDIS,NNODE,
     .                      SHAPE,STRES,IFFLE,IFSHE)
C***********************************************************************
C
C*** EVALUATES STRESSES FOR MINDLIN PLATE
C
C***********************************************************************
      DIMENSION CARTD(2,9),DFLEX(3,3),DGRAD(6),DSHER(2,2),
     .          ELDIS(3,9),SHAPE(9),STRES(5)
C*** ZERO STRESS VECTOR
      CALL      VZERO     (5,STRES)
C*** EVALUATE ROTATIONS AT GAUSS POINT , IF NEEDED
      IF(IFSHE.EQ.0) GOTO 50
      XZROT=0.0
      YZROT=0.0
      DO 30 INODE=1,NNODE
      XZROT=XZROT+SHAPE(INODE)*ELDIS(2,INODE)
   30 YZROT=YZROT+SHAPE(INODE)*ELDIS(3,INODE)
C*** EVALUATE BENDING STRESS RESULTANTS
   50 IF(IFFLE.EQ.0) GOTO 60
      EFLXX=-DGRAD(2)*CONST
      EFLYY=-DGRAD(6)*CONST
      EFLXY=-(DGRAD(3)+DGRAD(5))*CONST
      STRES(1)=DFLEX(1,1)*EFLXX+DFLEX(1,2)*EFLYY
      STRES(2)=DFLEX(2,1)*EFLXX+DFLEX(2,2)*EFLYY
      STRES(3)=DFLEX(3,3)*EFLXY
C*** EVALUATE SHEAR STRESS RESULTANTS
   60 IF(IFSHE.EQ.0) RETURN
      ESHXX=DGRAD(1)-XZROT
      ESHYY=DGRAD(4)-YZROT
      STRES(4)=DSHER(1,1)*ESHXX
      STRES(5)=DSHER(2,2)*ESHYY
      RETURN
      END
```

9.7 Examples

To test the program, the elasto-plastic analysis of a simply supported plate is performed and 9 noded and Heterosis elements are used. The geometry, material properties of the plate are shown in Fig. 9.6.



($L = 1.0$, $E = 10.92$, $\nu = 0.3$, $t = 0.01$, $q = 1.0$, $\sigma_0 = 1600.0$)

Fig. 9.6 Geometry and material properties of simply supported square plate.

Typical input for the nonlayered approach is given in Appendix IV together with lineprinter output of results. Figures 9.7 and 9.8 show the load displacement curves for both layered and nonlayered approaches.

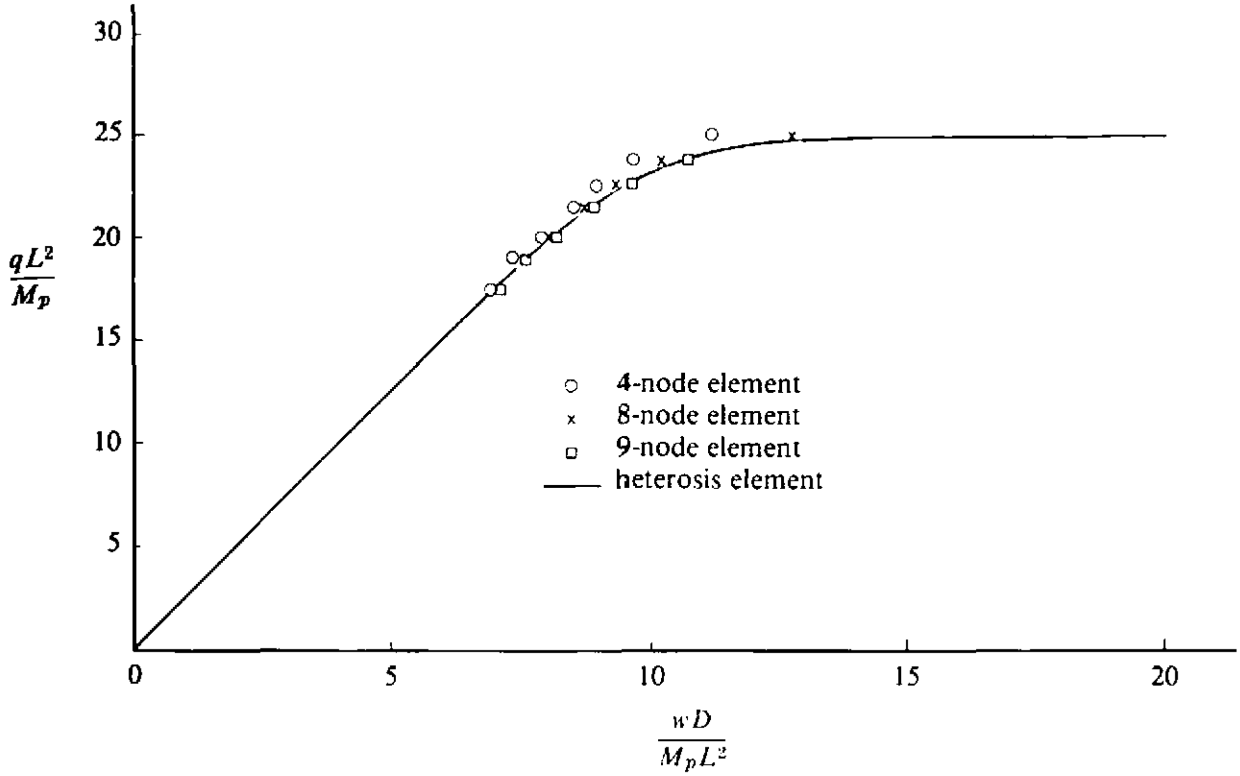


Fig. 9.7 Load displacement curves for nonlayered approach.

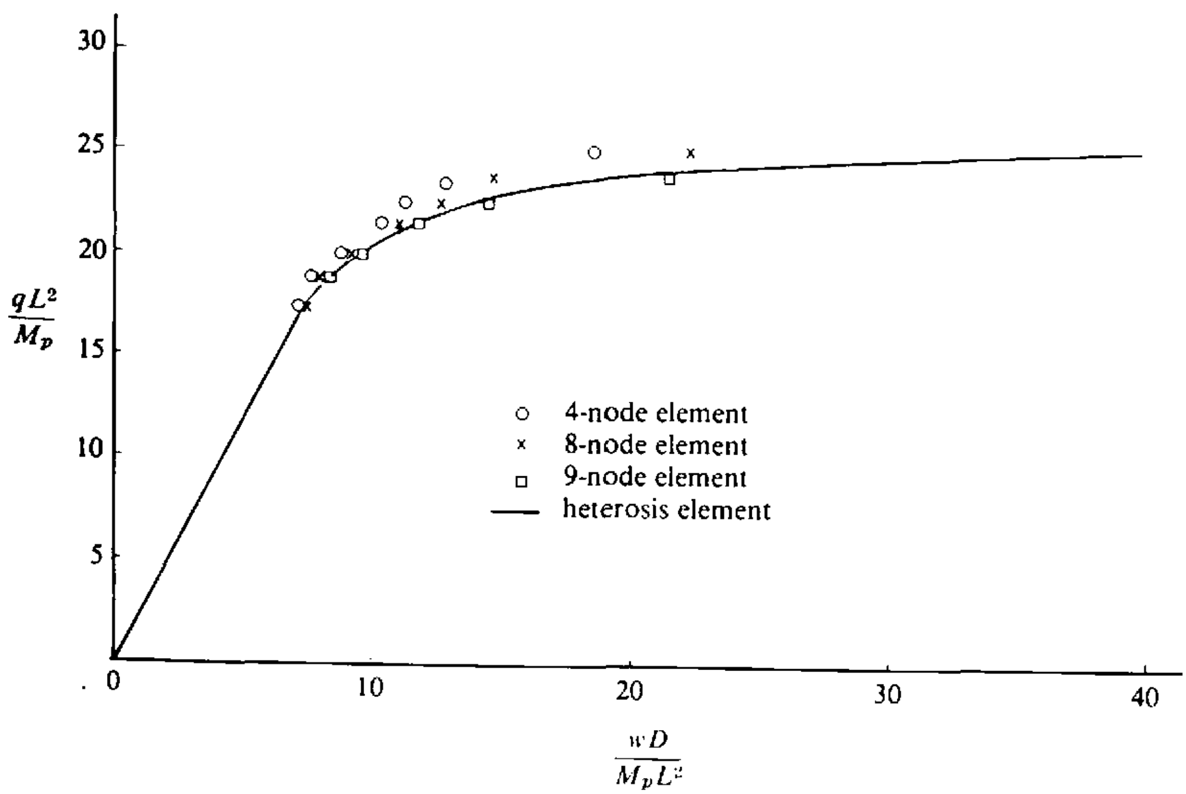


Fig. 9.8 Load displacement curves for layered approach.

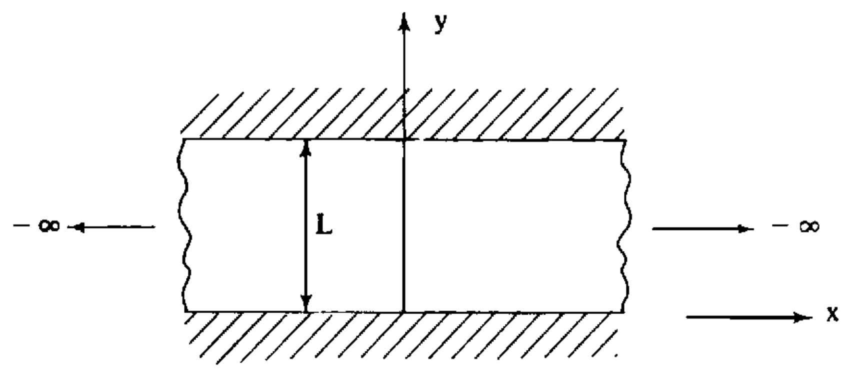


Fig. 9.9 Infinite clamped plate strip under uniform lateral load q.

9.8 Problems

9.1 Consider the uniformly loaded, clamped plate shown in Fig. 9.9. Using programs MINDLIN and MINDLAY find the collapse load for the plate which has the following properties:
Elastic modulus $E = 10000.0$, Poisson's ratio $\nu = 0.3$, thickness $t = 0.01$, length $L = 1.00$ and yield stress $\sigma_0 = 1000.0$. Check your solution using program PLANET.

9.2 Use program MINDLIN to find the value of the uniformly distributed load intensity q at which yielding first occurs for rectangular, simply supported plates of aspect ratios 1.0, 1.2, 1.4, 1.6, 2.0 and 2.2. Assume a thickness/span ratio of 0.05 and locate also the position of first yielding. Compare your results with those of Turvey[9] for a Von Mises material.

9.3 Modify program MINDLAY to allow for in-plane deformation of the plate mid-plane. Use a displacement pattern of the form

$$u(x, y, z) = u_0(x, y) - z\theta_x(x, y) \tag{9.31}$$

$$v(x, y, z) = v_0(x, y) - z\theta_y(x, y) \tag{9.32}$$

in which u_0 and v_0 are the in-plane deflections of the plate mid-plane in the x and y directions respectively.

9.4 Modify programs MINDLIN and MINDLAY to allow for an elastic Winkler foundation of modulus K. The appropriate virtual work term is

$$\int_{\Omega} \delta w \, K \, w \, d\Omega$$

in which δw is the virtual lateral displacement.

9.5 Solve the beam problem in Example 5.1 of Chapter 5 using programs MINDLIN and MINDLAY.

9.6 Develop a program for the nonlayered elastoplastic analysis of axisymmetric Mindlin plates using 2-node radial finite elements. The virtual work expression for an annular plate of internal and external radii r_0 and r_1 respectively is given as

$$2\pi \int_{r_0}^{r_1} \left[-\frac{d(\delta\theta)}{dr} M_r - \frac{\delta\theta}{r} M_\theta + \left(\frac{d(\delta w)}{dr} - \theta \right) Q \right] r\, dr$$

$$-2\pi \int_{r_0}^{r_1} \delta w q\, r\, dr \tag{9.33}$$

in which the radial bending moment $M_r = -D[d\theta/dr + \nu\,\theta/r]$ the circumferential bending moment $M_\theta = -D[\theta/r + \nu d\theta/dr]$ the shear force $Q = [Gt(dw/dr - \theta)]/1.2$, θ is the normal rotation in the radial rz plane and w is the lateral displacement in the z direction.

9.9 References

1. Hughes, T. J. R. and Cohen, M., The 'Heterosis' finite element for plate bending, *Computers and Structures* **9**, 445–450 (1978).
2. Bhaumik, A. K. and Hanley, J. T., Elasto-plastic plate analysis by finite differences, *J. Struct. Div. ASCE*, **93**, 575 (1967).
3. Armen, H., Pifko, A. and Levine, H. S., A finite element method for the plastic bending analysis of structures, *Proc. of the Second Conf. on Matrix Methods in Struct. Mech., Wright-Patterson Air Force Base*, Dayton, Ohio, 1301–1339 (1968).
4. Lopez, L. A. and Ang, A. H. I., Flexural analysis of elastic-plastic rectangular plates, *Civil Engineering Studies*, Structural Research Series No. 305, University of Illinois (1966).
5. McNiece, G. M. and Kemp, K. O., Comparison of finite element and unique limit analysis solutions for certain reinforced concrete slabs. *Proc. Instn. Civ. Engrs.* **43,** 629–640 (1969).
6. Backlund, J., Mixed finite element analysis of elastic and elasto-plastic plates in bending, *Chalmers University of Technology, Department of Structural Mechanics, Publication 71: 1*, **30,** Göteborg (1971).
7. Wegmuller, A. W. and Kostem, C. N., Finite element analysis of elastic-plastic plates and eccentrically stiffened plates, *Fritz Engineering Laboratory Report No.* **376A4**, Lehigh University, Bethlehem, Pennsylvania (1973).
8. Hinton, E. and Owen, D. R. J., *Finite Element Programming*, Academic Press (1977).
9. Turvey, G. J., First yield analysis of laterally loaded, rectangular, Levy plates with unsymmetric, side boundary conditions, *Proc. Instn. Civ. Engrs.*, **65,** 199–206 (1978).

Part III

Chapter 10
Explicit transient dynamic analysis

Written in collaboration with D. K. Paul and N. Bicanic

10.1 Introduction

Earlier, in Parts I and II, we considered static (or pseudostatic) applications. However, many structures are subjected to time-varying loads such as impulse, blast, impact or earthquake loading. Here in Part III we consider finite element based methods for dealing with such problems.

Although a form of mode-superposition has been adopted in nonlinear transient dynamic stress analysis,[1] it is general practice to use a time stepping procedure. Such direct integration schemes may be broadly classified as either explicit or implicit methods.

In the present chapter, we consider the very popular and easily implemented, explicit, central difference scheme. During each time step, relatively little computational effort is required since no formal matrix factorisation is necessary. Unfortunately, the method is conditionally stable and very small time steps are often needed.

In implicit schemes, a matrix factorisation is required but we can select an unconditionally stable implicit algorithm in which the time step length is governed by considerations of accuracy alone. In Chapter 11 we consider the Newmark family[2] of time stepping schemes. We then present a program for nonlinear transient dynamic stress analysis in which we may select any of the following algorithms:

(i) an implicit solution

(ii) an explicit solution

(iii) a combined implicit/explicit solution

The programs in Chapters 10 and 11 deal with plane stress, plane strain and axisymmetric applications using 4, 8 and 9-node, isoparametric quadrilaterals. Geometrically nonlinear behaviour is taken into account using a Total Lagrangian formulation. In Chapter 10 the material behaviour is assumed to be elasto-viscoplastic, whereas an elasto-plastic model is used in Chapter 11. Test examples are presented for both programs.

10.2 Dynamic equilibrium equations

For dynamic equilibrium of a body in motion we can use the Principle of Virtual Work to write the following equations at time station t_n irrespective of material behaviour

$$\int_{\Omega} [\delta \boldsymbol{\epsilon}_n]^T \boldsymbol{\sigma}_n \, d\Omega - \int_{\Omega} [\delta \boldsymbol{u}_n]^T [\boldsymbol{b}_n - \rho_n \ddot{\boldsymbol{u}}_n - c_n \dot{\boldsymbol{u}}_n] d\Omega - \int_{\Gamma_t} [\delta \boldsymbol{u}_n]^T \boldsymbol{t}_n \, d\Gamma = 0 \tag{10.1*}$$

where $\delta \boldsymbol{u}_n$ is the vector of virtual displacements, $\delta \boldsymbol{\epsilon}_n$ is the vector of associated virtual strains, $\boldsymbol{b}_n$ is the vector of applied body forces, $\boldsymbol{t}_n$ is the vector of surface tractions, $\boldsymbol{\sigma}_n$ is the vector of stresses, ρ_n is the mass density, c_n is the damping parameter and a dot refers to differentiation with respect to time. The domain of interest Ω has two boundaries: Γ_t on which boundary tractions $\boldsymbol{t}_n$ are specified and Γ_u on which displacements $\boldsymbol{u}_n$ are specified. For plane stress, plane strain and axisymmetric problems all of these terms were defined in Chapter 6.

Recall that in Chapter 6 we noted that, for a finite element representation, the displacements and strains and also their virtual counterparts are given by the relationships

$$\boldsymbol{u}_n = \sum_{i=1}^{m} \boldsymbol{N}_i [\boldsymbol{d}_i]_n, \qquad \delta \boldsymbol{u}_n = \sum_{i-1}^{m} \boldsymbol{N}_i [\delta \boldsymbol{d}_i]_n \tag{10.2}$$

$$\boldsymbol{\epsilon}_n = \sum_{i=1}^{m} \boldsymbol{B}_i [\boldsymbol{d}_i]_n, \qquad \delta \boldsymbol{\epsilon}_n = \sum_{i=1}^{m} \boldsymbol{B}_i [\delta \boldsymbol{d}_i]_n \tag{10.3}$$

where at time station t_n for node i, $[\boldsymbol{d}_i]_n$ is the vector of nodal displacements, $[\delta \boldsymbol{d}_i]_n$ is the vector of virtual nodal variables, $\boldsymbol{N}_i = N_i \boldsymbol{I}_2$ is the matrix of global shape functions and $\boldsymbol{B}_i$ is the global strain-displacement matrix.† The total number of nodes is m.

If (10.2) and (10.3) are substituted into (10.1), and if we note that the resulting equation is true for any set of virtual displacements $[\delta \boldsymbol{d}]_n$ then we obtain for each node i the equations.

* Note that a subscript n refers to a quantity sampled at time station t_n and similarly a subscript $n+1$ refers to a quantity sampled at time station $t_n + \Delta t$.

† Here we assume that the strains are linear and hence $\mathbf{B}_i$ is independent of time. Later we show how to cater for nonlinear strains in which $\mathbf{B}_i$ is displacement (and hence time) dependent and it is written as $[\mathbf{B}_i]_n$.

$$[\boldsymbol{p}_i]_n - [\boldsymbol{f}_{Bi}]_n + [\boldsymbol{f}_{Ii}]_n + [\boldsymbol{f}_{Di}]_n - [\boldsymbol{f}_{Ti}]_n = \mathbf{0} \tag{10.4}$$

where the internal resisting forces are

$$[\boldsymbol{p}_i]_n = \int_\Omega [\boldsymbol{B}_i]^T \boldsymbol{\sigma}_n \, d\Omega, \tag{10.5}$$

the consistent forces for the applied body forces are

$$[\boldsymbol{f}_{Bi}]_n = \int_\Omega [\boldsymbol{N}_i]^T \boldsymbol{b}_n \, d\Omega, \tag{10.6}$$

the inertia forces are

$$\begin{aligned} [\boldsymbol{f}_{Ii}]_n &= \int_\Omega [\boldsymbol{N}_i]^T \rho_n [\boldsymbol{N}_1, \boldsymbol{N}_2, \ldots, \boldsymbol{N}_m] d\Omega \begin{bmatrix} [\ddot{\boldsymbol{d}}_1]_n \\ [\ddot{\boldsymbol{d}}_2]_n \\ \vdots \\ [\ddot{\boldsymbol{d}}_m]_n \end{bmatrix} \\ &= \sum_{j=1}^{m} [\boldsymbol{M}_{ij}]_n [\ddot{\boldsymbol{d}}_j]_n, \end{aligned} \tag{10.7}$$

(N.B. $[\boldsymbol{M}_{ij}]_n$ is a submatrix of the mass matrix $\boldsymbol{M}_n$) The damping forces are

$$\begin{aligned} [\boldsymbol{f}_{Di}]_n &= \int_\Omega [\boldsymbol{N}_i]^T c_n [\boldsymbol{N}_1, \boldsymbol{N}_2, \ldots, \boldsymbol{N}_m] d\Omega \begin{bmatrix} [\dot{\boldsymbol{d}}_1] \\ [\dot{\boldsymbol{d}}_2] \\ \vdots \\ [\dot{\boldsymbol{d}}_m] \end{bmatrix} \\ &= \sum_{j=1}^{m} [\boldsymbol{C}_{ij}]_n [\dot{\boldsymbol{d}}_j]_n \end{aligned} \tag{10.8}$$

(N.B. $[\boldsymbol{C}_{ij}]_n$ is a submatrix of the damping matrix $\boldsymbol{C}_n$) and the consistent forces for the traction boundary forces are

$$[\boldsymbol{f}_{Ti}]_n = \int_{\Gamma_t} [\boldsymbol{N}_i]^T \boldsymbol{t}_n \, d\Gamma. \tag{10.9}$$

If we use $C(0)$ isoparametric finite element representations we can evaluate contributions to (10.4) separately from each element and then assemble them into the appropriate vectors in (10.4). As noted in Chapter 6 the displacements can be expressed in the usual way as

$$[\boldsymbol{u}^{(e)}]_n = \sum_{i=1}^{r} \boldsymbol{N}_i^{(e)} [\boldsymbol{d}_i^{(e)}]_n \tag{10.10}$$

where for local node i of element e, $\boldsymbol{N}_i^{(e)} = N_i^{(e)} \boldsymbol{I}_2$ is the local shape function matrix and $[\boldsymbol{d}_i^{(e)}]_n$ is the vector of nodal displacements. As described in

Chapter 6 we use 4, 8 and 9 noded isoparametric quadrilateral elements and therefore $r = 4$, 8 and 9 respectively for these cases.

The strain displacement relationships are expressed as

$$[\boldsymbol{\epsilon}^{(e)}]_n = \sum_{i=1}^{r} \boldsymbol{B}_i^{(e)}[\boldsymbol{d}_i^{(e)}]_n \tag{10.11}$$

in which $\boldsymbol{B}_i^{(e)}$ is the local element strain matrix which has been defined for the various applications in Table 6.1.

The discretised elemental volume is given as

$$d\Omega^{(e)} = h^{(e)} \det \boldsymbol{J}^{(e)} d\xi d\eta \tag{10.12}$$

in which $\det \boldsymbol{J}^{(e)}$ is the determinant of the Jacobian matrix and $h^{(e)}$ is defined in Chapter 6.

Thus the element contributions to the terms in (10.4) may be evaluated using numerical integration based on Gauss–Legendre product rules. These contributions now take the form

$$[\boldsymbol{p}_i^{(e)}]_n = \int_{-1}^{+1}\int_{-1}^{+1} [\boldsymbol{B}_i^{(e)}]^T \boldsymbol{\sigma}_n^{(e)} h^{(e)} \det \boldsymbol{J}^{(e)} d\xi\, d\eta \tag{10.13}$$

$$[\boldsymbol{f}_{Bi}^{(e)}]_n = \int_{-1}^{+1}\int_{-1}^{+1} [\boldsymbol{N}_i^{(e)}]^T \boldsymbol{b}_n h^{(e)} \det \boldsymbol{J}^{(e)} d\xi\, d\eta \tag{10.14}$$

$$\begin{aligned}[\boldsymbol{f}_{Ii}^{(e)}]_n &= \int_{-1}^{+1}\int_{-1}^{+1} [\boldsymbol{N}_i^{(e)}]^T \rho_n^{(e)} [\boldsymbol{N}_1^{(e)}, \boldsymbol{N}_2^{(e)}, \ldots, \boldsymbol{N}_r^{(e)}] h^{(e)} \det \boldsymbol{J}^{(e)} d\xi\, d\eta \begin{bmatrix} [\ddot{\boldsymbol{d}}_1^{(e)}]_n \\ \vdots \\ [\ddot{\boldsymbol{d}}_r^{(e)}]_n \end{bmatrix} \\ &= \sum_{j=1}^{r} [\boldsymbol{M}_{ij}^{(e)}]_n [\ddot{\boldsymbol{d}}_j^{(e)}]_n \end{aligned} \tag{10.15}$$

$$\begin{aligned}[\boldsymbol{f}_{Di}^{(e)}]_n &= \int_{-1}^{+1}\int_{-1}^{+1} [\boldsymbol{N}_i^{(e)}]^T c_n^{(e)} [\boldsymbol{N}_1^{(e)}, \boldsymbol{N}_2^{(e)}, \ldots, \boldsymbol{N}_r^{(e)}] h^{(e)} \det \boldsymbol{J}^{(e)} d\xi\, d\eta \begin{bmatrix} [\dot{\boldsymbol{d}}_1^{(e)}]_n \\ \vdots \\ [\dot{\boldsymbol{d}}_r^{(e)}]_n \end{bmatrix} \\ &= \sum_{j=1}^{r} [\boldsymbol{C}_{ij}^{(e)}]_n [\dot{\boldsymbol{d}}_j^{(e)}]_n \end{aligned} \tag{10.16}$$

$$[\boldsymbol{f}_{Ti}^{(e)}]_n = \int_{\Gamma_t^{(e)}} [\boldsymbol{N}_i^{(e)}]^T \boldsymbol{t}_n^{(e)} d\Gamma \tag{10.17}$$

where $\Gamma_t^{(e)}$ (if it exists) is that part of Γ_t which coincides with the boundary of element domain $\Omega^{(e)}$.

We will assume for simplicity that the mass and damping matrices do not vary with time.

10.3 Modelling of nonlinearities

10.3.1 Introduction

Dynamic loading of structures often causes excursions of stresses well into the inelastic range and the influence of geometry changes on the response is also significant in many cases. Therefore both material and geometric non-linear effects should be considered.

Although material behaviour under dynamic loading is very complex and experimental information is scarce, for most structural materials, some general statements can be made.

For example, it has frequently been demonstrated that the instantaneous yield stress is significantly influenced by the rate of straining. Also, the value of the elasticity modulus E_0 is found to be dependent on the strain rate. For structural materials with limited ductility, such as concrete or rock-like materials, the rate of straining can completely change the material response from elasto-plastic behaviour under low rates to brittle elastic behaviour under high rates of straining. For many structural materials there is still an urgent need for a better understanding of the observed phenomena and underlying microscopic behaviour. However, in attempting to perform an analysis of a dynamically-loaded engineering structure, we must look for an idealized material model, where possibly some compromises have to be made. Furthermore, the model parameters should readily be measurable and easily obtained from reliable experimental data.

For transient dynamic analysis, an elasto-viscoplastic model, as developed in earlier chapters, presents a very good approximation of the true behaviour for many structural materials. The predominant phenomenon of variable instantaneous yield stress is adequately modelled.

In the following, we shall develop the algorithm for the elasto-viscoplastic transient dynamic analysis of plane stress, plane strain and axisymmetric problems. The computer program DYNPAK will be documented and explained and finally, some illustrative examples are given.

10.3.2 Material model

Here we adopt the elasto-viscoplastic material model developed in Chapter 8, where the constitutive relationship is given in the form

$$\begin{aligned}\dot{\boldsymbol{\epsilon}}_n &= [\dot{\boldsymbol{\epsilon}}_e]_n + [\dot{\boldsymbol{\epsilon}}_{vp}]_n \\ &= [\boldsymbol{D}]^{-1}\dot{\boldsymbol{\sigma}}_n + \gamma\langle\Phi_n(F)\rangle\frac{\partial F}{\partial \boldsymbol{\sigma}_n}\end{aligned} \tag{10.18}$$

where $\boldsymbol{D}$ is the elasticity matrix, γ is the fluidity parameter, F is the yield function and $\dot{\epsilon}_n$, $[\dot{\epsilon}_e]_n$ and $[\dot{\epsilon}_{vp}]_n$ denote the total, elastic and viscoplastic strain rates at time station t_n. We also have the relationships

$$\begin{aligned} \sigma_n &= D[\epsilon_e]_n \\ \epsilon_n &= [\epsilon_e]_n + [\epsilon_{vp}]_n \end{aligned} \tag{10.19}$$

and

$$\begin{aligned} \langle \Phi_n(F) \rangle &= 0 \quad \text{if yield has not occurred.} \\ &= 1 \quad \text{if yield has occurred.} \end{aligned} \tag{10.20}$$

Thus we can rewrite the internal resisting forces as

$$p_n = \int_\Omega [B]^T D\{\epsilon_n - [\epsilon_{vp}]_n\} d\Omega \tag{10.21}$$

The temporal discretization of the equations which govern viscoplastic straining is also based on the assumption that the relationship

$$[\dot{\epsilon}_{vp}]_n = \gamma \langle \Phi_n(F) \rangle \frac{\partial F}{\partial \sigma_n} \tag{10.22}$$

is known only for discrete time stations Δt apart. The simplest, Euler, integration scheme will here be employed, i.e.,

$$[\epsilon_{vp}]_{n+1} = [\epsilon_{vp}]_n + [\dot{\epsilon}_{vp}]_n \Delta t. \tag{10.23*}$$

The stability limit for the time increment Δt, which depends on the specific form of the viscoplastic potential employed in the flow rule, has already been discussed in earlier chapters.

When we adopt the central difference scheme and the viscoplastic material model that we have just described, the algorithm at a particular time station t_n follows the sequence shown in Fig. 10.1.

10.3.3 Geometric nonlinearity

If we wish to cater for geometrically nonlinear elastic behaviour we can choose either a total or updated Lagrangian coordinate system. Here we choose a total Lagrangian coordinate system which coincides with the initial undeformed position of the body.(3)

It transpires that, with the central difference scheme, the only changes required to account for geometrically nonlinear effects are

(i) The modification of the strain-displacement matrix $B(d_n)$,

and

(ii) The evaluation of the strains using a deformation Jacobian matrix $J_D(d_n)$.

* Note that in dynamic transient analysis, the time interval Δt is here assumed constant; whereas for viscoplastic applications in Chapter 8 it is variable.

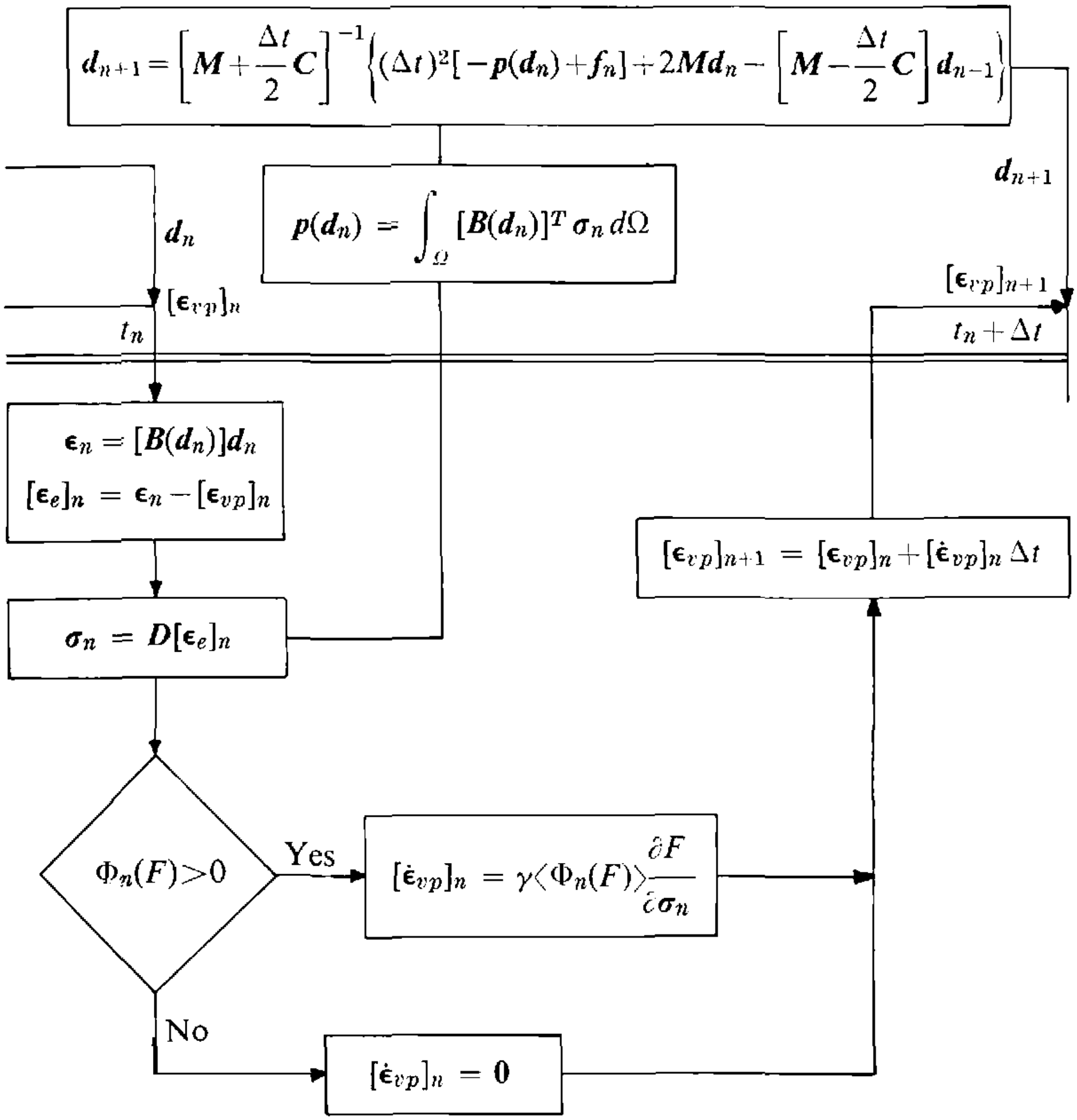


Fig. 10.1 Algorithm for elasto viscoplastic straining during a time step.

We will now describe briefly the relevant background theory. All vectors and matrices are given explicitly for the plane stress, plane strain and axisymmetric applications in Table 10.1.

If the initial undeformed position of a particle of material is x_0 and the total displacement vector at time station t_n is u_n then the coordinates of the particle are

$$x_n = x_0 + u_n \tag{10.24}$$

In a total Lagrangian formulation we use Green's strains. The matrix of Green's strains is given as

$$E_n = \tfrac{1}{2}\left[[J_D]_n{}^T [J_D]_n - I\right] \tag{10.25}$$

Table 10.1 Vectors and matrices used in a total Lagrangian formulation

Variables	Plane stress/strain	Axisymmetric
Coordinates of particle in undeformed initial configuration $\boldsymbol{x} = \boldsymbol{x}_0$	$[x_0, y_0]^T$	$[r_0, z_0]^T$
Displacements $\boldsymbol{u}_n$	$[u_n, v_n]^T$	$[u_n, w_n]^T$
Coordinates of particle in deformed configuration $\boldsymbol{x}_n$	$[x_n, y_n]^T = [x_0+u_n, y_0+v_n]^T$	$[r_n, z_n]^T = [r_0+u_n, z_0+w_n]$
Vector of Green's strains $\boldsymbol{\epsilon}_n$	$\begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{bmatrix}_n = \begin{bmatrix} \dfrac{\partial u_n}{\partial x} + \dfrac{1}{2}\left(\dfrac{\partial u_n}{\partial x}\right)^2 + \dfrac{1}{2}\left(\dfrac{\partial v_n}{\partial x}\right)^2 \\ \dfrac{\partial v_n}{\partial y} + \dfrac{1}{2}\left(\dfrac{\partial u_n}{\partial y}\right)^2 + \dfrac{1}{2}\left(\dfrac{\partial v_n}{\partial y}\right)^2 \\ \dfrac{\partial u_n}{\partial y}+\dfrac{\partial v_n}{\partial x}+\dfrac{\partial u_n}{\partial x}\dfrac{\partial u_n}{\partial y}+\dfrac{\partial v_n}{\partial x}\dfrac{\partial v_n}{\partial y} \end{bmatrix}$	$\begin{bmatrix} \epsilon_r \\ \epsilon_z \\ \gamma_{rz} \\ \epsilon_\theta \end{bmatrix}_n = \begin{bmatrix} \dfrac{\partial u_n}{\partial r} + \dfrac{1}{2}\left(\dfrac{\partial u_n}{\partial r}\right)^2 + \dfrac{1}{2}\left(\dfrac{\partial w_n}{\partial r}\right)^2 \\ \dfrac{\partial w_n}{\partial z} + \dfrac{1}{2}\left(\dfrac{\partial u_n}{\partial z}\right)^2 + \dfrac{1}{2}\left(\dfrac{\partial w_n}{\partial z}\right)^2 \\ \dfrac{\partial u_n}{\partial z}+\dfrac{\partial w_n}{\partial r}+\dfrac{\partial u_n}{\partial r}\dfrac{\partial u_n}{\partial z}+\dfrac{\partial w_n}{\partial r}\dfrac{\partial w_n}{\partial z} \\ \dfrac{u_n}{r}+\dfrac{1}{2}\left(\dfrac{u_n}{r}\right)^2 \end{bmatrix}$
Deformation Jacobian matrix $\boldsymbol{J}_D(\boldsymbol{u}_n) = [\boldsymbol{J}_D]_n$	$\begin{bmatrix} \dfrac{\partial x_n}{\partial x} & \dfrac{\partial x_n}{\partial y} \\ \dfrac{\partial y_n}{\partial x} & \dfrac{\partial y_n}{\partial y} \end{bmatrix}$	$\begin{bmatrix} \dfrac{\partial r_n}{\partial r} & \dfrac{\partial r_n}{\partial z} \\ \dfrac{\partial z_n}{\partial r} & \dfrac{\partial z_n}{\partial z} \end{bmatrix}$
Matrix of Green's strains $\boldsymbol{E}_n = \frac{1}{2}\{[\boldsymbol{J}_D]_n{}^T[\boldsymbol{J}_D]_n - \boldsymbol{I}\}$	$\begin{bmatrix} \epsilon_{xx} & \epsilon_{xy} \\ \epsilon_{yx} & \epsilon_{yy} \end{bmatrix}_n$	$\begin{bmatrix} \epsilon_{rr} & \epsilon_{rz} \\ \epsilon_{zr} & \epsilon_{zz} \end{bmatrix}_n$
Linear strains $[\boldsymbol{\epsilon}_L]_n$	$\left[\dfrac{\partial u_n}{\partial x}, \dfrac{\partial v_n}{\partial y}, \left(\dfrac{\partial u_n}{\partial y}+\dfrac{\partial v_n}{\partial x}\right)\right]^T$	$\left[\dfrac{\partial u_n}{\partial r}, \dfrac{\partial w_n}{\partial r}, \dfrac{\partial u_n}{\partial z}+\dfrac{\partial w_n}{\partial r}, \dfrac{u_n}{r}\right]^T$

Table 10.1 (Cont.)

Variable	Plane stress/strain	Axisymmetric
Nonlinear strains $[\boldsymbol{\epsilon}_{NL}]_n = \frac{1}{2}[\boldsymbol{A}_\theta]_n \boldsymbol{\theta}_n$ where $[\boldsymbol{A}_\theta]_n$ is	$\begin{bmatrix} \frac{\partial u_n}{\partial x} & \frac{\partial v_n}{\partial x} & 0 & 0 \\ 0 & 0 & \frac{\partial u_n}{\partial y} & \frac{\partial v_n}{\partial y} \\ \frac{\partial u_n}{\partial y} & \frac{\partial v_n}{\partial y} & \frac{\partial u_n}{\partial x} & \frac{\partial v_n}{\partial x} \end{bmatrix}$	$\begin{bmatrix} \frac{\partial u_n}{\partial r} & \frac{\partial w_n}{\partial r} & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial u_n}{\partial z} & \frac{\partial w_n}{\partial z} & 0 \\ \frac{\partial u_n}{\partial z} & \frac{\partial w_n}{\partial z} & \frac{\partial u_n}{\partial r} & \frac{\partial w_n}{\partial r} & 0 \\ 0 & 0 & 0 & 0 & \frac{u_n}{r} \end{bmatrix}$
and displacement gradients $\boldsymbol{\theta}_n$	$\begin{bmatrix} \frac{\partial u_n}{\partial x} & 0 & \frac{\partial u_n}{\partial x} \\ \frac{\partial v_n}{\partial x} & 0 & \frac{\partial v_n}{\partial x} \\ 0 & \frac{\partial u_n}{\partial y} & \frac{\partial u_n}{\partial y} \\ 0 & \frac{\partial v_n}{\partial y} & \frac{\partial v_n}{\partial y} \end{bmatrix}$	$\begin{bmatrix} \frac{\partial u_n}{\partial r} & 0 & \frac{\partial u_n}{\partial r} & 0 \\ \frac{\partial w_n}{\partial r} & 0 & \frac{\partial w_n}{\partial r} & 0 \\ 0 & \frac{\partial u_n}{\partial z} & \frac{\partial u_n}{\partial z} & 0 \\ 0 & \frac{\partial w_n}{\partial z} & \frac{\partial w_n}{\partial z} & 0 \\ 0 & 0 & 0 & \frac{u_n}{r} \end{bmatrix}$
Elastic Piola–Kirchoff stresses $\boldsymbol{\sigma}_n = \boldsymbol{D}_n \boldsymbol{\epsilon}_n$	$[\sigma_x, \quad \sigma_y, \quad \tau_{xy}]_n{}^T$	$[\sigma_r, \quad \sigma_z, \quad \tau_{rz}, \quad \sigma_\theta]_n{}^T$

where $[\boldsymbol{J}_D]_n$ is the deformation Jacobian matrix at time station t_n.

The Green's strains can be written as

$$\boldsymbol{\epsilon}_n = [\boldsymbol{\epsilon}_L]_n + [\boldsymbol{\epsilon}_{NL}]_n \tag{10.26}$$

where $[\boldsymbol{\epsilon}_L]_n$ are the linear strains given earlier in Chapter 6 and $[\boldsymbol{\epsilon}_{NL}]_n$, the nonlinear strain terms are given as

$$[\boldsymbol{\epsilon}_{NL}]_n = \tfrac{1}{2}[\boldsymbol{A}_\theta]_n \boldsymbol{\theta}_n. \tag{10.27}$$

For a set of virtual displacements, the corresponding virtual Green's strains are given as

$$[\delta\boldsymbol{\epsilon}]_n = [\delta\boldsymbol{\epsilon}_L]_n + [\boldsymbol{A}_\theta]_n \,\delta\boldsymbol{\theta}_n. \tag{10.28}$$

Thus the virtual work statement of (10.1) can be rewritten as

$$\int_\Omega [\delta\boldsymbol{\epsilon}_n]^T \boldsymbol{\sigma}_n \, d\Omega - \int_\Omega [\delta\boldsymbol{u}_n]^T [\boldsymbol{b}_n - \rho\ddot{\boldsymbol{u}}_n - c\dot{\boldsymbol{u}}_n] d\Omega - \int_{\Gamma_t} [\delta\boldsymbol{u}_n]^T \boldsymbol{t}_n \, d\Gamma = 0 \tag{10.29}$$

where $\boldsymbol{\sigma}_n$ are the Piola–Kirchhoff stresses.

As mentioned earlier, all relevant terms are given in Table 10.1.

If we adopt the finite element discretization scheme described earlier, then the displacement gradients $\boldsymbol{\theta}_n$ are given in terms of the nodal displacements $[\boldsymbol{d}_i]_n$ by the linear relation

$$\boldsymbol{\theta}_n = \sum_{i=1}^{m} \boldsymbol{G}_i [\boldsymbol{d}_i]_n \tag{10.30}$$

where $\boldsymbol{G}_i$ contains Cartesian shape function derivatives as indicated in Table 10.2 for the various applications.

Similarly we have

$$\delta\boldsymbol{\theta}_n = \sum_{i=1}^{m} \boldsymbol{G}_i [\delta\boldsymbol{d}_i]_n. \tag{10.31}$$

The linear strain-displacement relationship can be expressed as

$$[\boldsymbol{\epsilon}_L]_n = \sum_{i=1}^{m} [\boldsymbol{B}_{Li}]_n [\boldsymbol{d}_i]_n \tag{10.32}$$

where $[\boldsymbol{B}_{Li}]_n$ is the linear strain displacement matrix introduced earlier.

Table 10.2 The nonlinear strain displacement matrix evaluation in a total Lagrangian finite element formulation

Variable	Plane stress/strain	Axisymmetric
Strain displacement matrix associated with node i $[\boldsymbol{B}_i]_n = [\boldsymbol{B}_{Li}]_n + [\boldsymbol{A}_0]_n \boldsymbol{G}_i$	$\begin{bmatrix} \dfrac{\partial x_n}{\partial x}\dfrac{\partial N_i}{\partial x} & \dfrac{\partial y_n}{\partial x}\dfrac{\partial N_i}{\partial x} \\ \dfrac{\partial x_n}{\partial y}\dfrac{\partial N_i}{\partial y} & \dfrac{\partial y_n}{\partial y}\dfrac{\partial N_i}{\partial y} \\ \left(\dfrac{\partial x_n}{\partial y}\dfrac{\partial N_i}{\partial x}+\dfrac{\partial x_n}{\partial x}\dfrac{\partial N_i}{\partial y}\right) & \left(\dfrac{\partial y_n}{\partial y}\dfrac{\partial N_i}{\partial x}+\dfrac{\partial y_n}{\partial x}\dfrac{\partial N_i}{\partial y}\right) \end{bmatrix}$	$\begin{bmatrix} \dfrac{\partial r_n}{\partial r}\dfrac{\partial N_i}{\partial r} & \dfrac{\partial z_n}{\partial r}\dfrac{\partial N_i}{\partial r} \\ \dfrac{\partial r_n}{\partial z}\dfrac{\partial N_i}{\partial z} & \dfrac{\partial z_n}{\partial z}\dfrac{\partial N_i}{\partial z} \\ \left(\dfrac{\partial r_n}{\partial z}\dfrac{\partial N_i}{\partial r}+\dfrac{\partial r_n}{\partial r}\dfrac{\partial N_i}{\partial z}\right) & \dfrac{\partial z_n}{\partial z}\dfrac{\partial N_i}{\partial r}+\dfrac{\partial z_n}{\partial r}\dfrac{\partial N_i}{\partial z} \\ \left(\dfrac{r_n}{r}\right)\dfrac{N_i}{r} & 0 \end{bmatrix}$
where $\boldsymbol{G}_i$ is	$\begin{bmatrix} \dfrac{\partial N_i}{\partial x} & 0 & \dfrac{\partial N_i}{\partial y} & 0 \\ 0 & \dfrac{\partial N_i}{\partial x} & 0 & \dfrac{\partial N_i}{\partial y} \end{bmatrix}$	$\begin{bmatrix} \dfrac{\partial N_i}{\partial r} & 0 & \dfrac{\partial N_i}{\partial z} & 0 & \dfrac{N_i}{r} \\ 0 & \dfrac{\partial N_i}{\partial r} & 0 & \dfrac{\partial N_i}{\partial z} & 0 \end{bmatrix}$

Similarly, we have

$$[\delta\epsilon_{NL}]_n = \sum_{i=1}^{m} [B_{NLi}]_n[\delta d_i]_n \tag{10.33}$$

The components of the vector of Green's strains ϵ_n can be written as

$$\epsilon_n = \sum_{i=1}^{m} \left[[B_{Li}]_n + \tfrac{1}{2}[B_{NLi}]_n\right] [d_i]_n \tag{10.34}$$

where the nonlinear strain-displacement matrix $[B_{NLi}]_n$ is given as

$$[B_{NLi}]_n = [A_\theta]_n G_i. \tag{10.35}$$

Furthermore it can be shown that the virtual strains can be expressed as

$$\delta\epsilon_n = \sum_{i=1}^{m} [B_i]_n[\delta d_i]_n \tag{10.36}$$

where

$$[B_i]_n = [B_{Li}]_n + [B_{NLi}]_n$$

is given in Table 10.2 for the various applications.

If we substitute for $\delta\epsilon_n$ and δd_n in (10.29) and note that the result is true for arbitrary virtual displacements, then we obtain an expression which is identical to (10.4). In the present case we only need to remember that $[B_i]_n$ is defined by (10.36).

We again note that contributions to (10.4) from each element can be obtained separately and assembled appropriately.

Note that we now may evaluate $[p_i]_n$ as

$$\int_\Omega [B_i]_n^T \sigma_n \, d\Omega \quad \text{rather than} \quad \int_\Omega [B_i]^T \sigma_n \, d\Omega$$

where $[B_i]_n$ is given by (10.36).

10.4 Explicit time integration scheme

10.4.1 Central difference approximation

We can write the equations (10.4) in matrix form so that at time station t_n we have

$$M\ddot{d}_n + C\dot{d}_n + p_n = f_n \tag{10.37}^*$$

* Note that the body force term $-\mathbf{M}\ddot{\mathbf{u}}_g$, due to seismic excitation, is included in the body forces which are taken into account in $\mathbf{f}_n$. Note also that $\mathbf{M}$ and $\mathbf{C}$ may be assembled from the element mass matrices $\mathbf{M}^{(e)}$ and damping matrices $\mathbf{C}^{(e)}$.

where $\boldsymbol{M}$ and $\boldsymbol{C}$ are the global mass and damping matrices respectively, $\boldsymbol{p}_n$ is the global vector of internal resisting nodal forces, $\boldsymbol{f}_n$ is the vector of consistent nodal forces for the applied body and surfaces traction forces grouped together, $\ddot{\boldsymbol{d}}_n$ is the global vector of nodal accelerations and $\dot{\boldsymbol{d}}_n$ is the global vector of nodal velocities.

So far, only spatial discretization has been introduced. We now employ a temporal discretization of the dynamic equilibrium equations by approximating the accelerations and velocities using finite difference expressions.

In particular we adopt a central difference approximation[2] so that the accelerations can be written as

$$\ddot{\boldsymbol{d}}_n \simeq \boldsymbol{a}_n = \frac{1}{(\Delta t)^2}\{\boldsymbol{d}_{n+1} - 2\boldsymbol{d}_n + \boldsymbol{d}_{n-1}\} \tag{10.38}$$

and the velocities are written as

$$\dot{\boldsymbol{d}}_n \simeq \boldsymbol{v}_n = \frac{1}{2\Delta t}\{\boldsymbol{d}_{n+1} - \boldsymbol{d}_{n-1}\} \tag{10.39}$$

in which Δt is the time step or interval so that we are sampling the displacements at time stations $t_n - \Delta t$, t_n and $t_n + \Delta t$. If we substitute (10.38) and (10.39) into (10.37) we obtain

$$\boldsymbol{M}\left\{\frac{\boldsymbol{d}_{n+1} - 2\boldsymbol{d}_n + \boldsymbol{d}_{n-1}}{(\Delta t)^2}\right\} + \boldsymbol{C}\left\{\frac{\boldsymbol{d}_{n+1} - \boldsymbol{d}_{n-1}}{2\Delta t}\right\} + \boldsymbol{p}_n = \boldsymbol{f}_n \tag{10.40}$$

which can be rearranged to give

$$\begin{aligned}\boldsymbol{d}_{n+1} = {} & \left[\boldsymbol{M} + \frac{\Delta t}{2}\boldsymbol{C}\right]^{-1} \\ & \times\left\{(\Delta t)^2\left[-\boldsymbol{p}_n + \boldsymbol{f}_n\right] + 2\boldsymbol{M}\boldsymbol{d}_n - \left[\boldsymbol{M} - \frac{\Delta t}{2}\boldsymbol{C}\right]\boldsymbol{d}_{n-1}\right\}.\end{aligned} \tag{10.41}$$

Thus we have

$$\boldsymbol{d}_{n+1} = \boldsymbol{g}(\boldsymbol{d}_n, \boldsymbol{d}_{n-1}). \tag{10.42}$$

In other words the displacements at time station $t_n + \Delta t$ are given explicitly in terms of the displacements at time stations t_n and $t_n - \Delta t$.

If the mass matrix $\boldsymbol{M}$ and the damping matrix $\boldsymbol{C}$ are diagonal then the solution of (10.41) becomes trivial and we have for plane stress and plane strain applications the following equations:

$$\begin{aligned}(d_{ui})_{n+1} = {} & \left(m_{ii} + \frac{\Delta t}{2}c_{ii}\right)^{-1}\Big[(\Delta t)^2\{-(p_{ui})_n + (f_{ui})_n\} \\ & + 2m_{ii}(d_{ui})_n - \left(m_{ii} - \frac{\Delta t}{2}c_{ii}\right)(d_{ui})_{n-1}\Big]\end{aligned} \tag{10.43}$$

and

$$(d_{vi})_{n+1} = \left(m_{ii}+\frac{\Delta t}{2}c_{ii}\right)^{-1}\left[(\Delta t)^2\{-(p_{vi})_n+(f_{vi})_n\}\right.$$
$$\left.+2m_{ii}(d_{vi})_n-\left(m_{ii}-\frac{\Delta t}{2}c_{ii}\right)(d_{vi})_{n-1}\right] \tag{10.44}$$

in which at node i, d_{ui} and d_{vi} are the u and v displacement components in the x and y directions, f_{ui} and f_{vi} are the components of the applied nodal forces in the x and y directions, p_{ui} and p_{vi} are the internal resisting nodal forces in the x and y directions and m_{ii} and c_{ii} are the diagonal terms of the mass and damping matrices. For axisymmetric problems replace v by w.

From (10.43) and (10.44) we see that for each displacement degree of freedom at time $t_n+\Delta t$ we have a separate equation involving information regarding the degree of freedom at times t_n and $t_n-\Delta t$. No matrix factorisation or sophisticated equation solving is therefore necessary.

10.4.2 Starting algorithm

As we have seen the governing equilibrium equation at time station $t_n+\Delta t$ in the central difference method involves information at the two previous time stations t_n and $t_n-\Delta t$. A starting algorithm is therefore necessary and from the initial conditions the values $\boldsymbol{d}(0-\Delta t)$ may be obtained. We have from (10.39) the condition that

$$\dot{\boldsymbol{d}}(0) \simeq \boldsymbol{v}(0) = \frac{\boldsymbol{d}(0+\Delta t)-\boldsymbol{d}(0-\Delta t)}{2\Delta t} \tag{10.45}$$

or

$$\boldsymbol{d}(0-\Delta t) = -2\Delta t v(0)+\boldsymbol{d}(0+\Delta t).$$

If this approximation is substituted in (10.43) then we can write the expression

$$(d_{ui})_1 = \left(m_{ii}+\frac{\Delta t}{2}c_{ii}\right)^{-1}\left[(\Delta t)^2\{-(p_{ui})_0+(f_{ui})_0\}\right.$$
$$\left.+2m_{ii}(d_{ui})_0-\left(m_{ii}-\frac{\Delta t}{2}c_{ii}\right)\{-2\Delta t(\dot{d}_{ui})_0+d_{ui})_1\}\right] \tag{10.46}$$

or

$$(d_{ui})_1 = \frac{(\Delta t)^2}{2m_{ii}}\{-(p_{ui})_0+(f_{ui})_0\}+(\dot{d}_{ui})_0+B\Delta t(d_{ui})_0$$

where

$$B = 1-\frac{c_{ii}\Delta t}{2m_{ii}}.$$

10.4.3 Damping

Very limited information is available on damping in linear solid mechanics problems and there is even less data available for damping in nonlinear situations. It is therefore customary to assume that the damping matrix is proportional to the mass and stiffness matrix. This is known as Rayleigh damping and we have

$$\boldsymbol{C} = \alpha\boldsymbol{M}+\beta\boldsymbol{K} \tag{10.47}$$

In the central difference method we can make the approximation that $\beta = 0$ so that

$$\boldsymbol{C} = \alpha\boldsymbol{M} \tag{10.48}$$

or

$$c_{ii} = \alpha m_{ii}$$

where

$$\alpha = 2\xi_r\omega_r$$

in which ξ_r and ω_r are the damping factor and circular frequency for the r^{th} mode. This modelling of damping is rather poor since α is fixed for all modes of vibration. Thus if we take $r = 1$ then the higher modes will be less damped whereas the opposite would be more desirable. This is the price we pay for an otherwise convenient and efficient solution.

10.5 Critical time step

In explicit and implicit time integration schemes the selection of an appropriate time step is crucially important. Small time steps are required for accurate and stable solutions whereas for reasons of economy we would prefer large time steps. The analysis of the stability and accuracy characteristics[2] allows us to decide on a suitable time step for the various time stepping schemes. On this basis for the conditionally stable, central difference scheme, the stability considerations are of prime importance and the time step length is limited by the expression

$$\Delta t \leqslant \frac{2}{\omega_{\max}} \tag{10.49}$$

where $\omega_{\max}$ is the highest circular frequency of the finite element mesh. This severe time step limit, required for stability, ensures accuracy in practically all modes of vibration. Providing that $\omega_{\max}$ represents the maximum nonlinear frequency, (10.49) holds for nonlinear problems. The estimate of the critical time step for conditionally stable schemes apparently necessitates the solution of the eigenvalue problem for the whole system. This is not so. The bound on the highest eigenvalue can be simply obtained by the consideration of an individual element. This is established by an important theorem proposed by Irons[4] which proves that the highest system eigenvalue must always be less than the highest eigenvalue of the individual elements. This allows a very easy estimate of critical time steps (by the above theorem) which will err on the safe side. To avoid the exact evaluation of the highest finite element mesh frequency approximate expressions are usually employed. The most common form for plane strain is

$$\Delta t \leqslant \mu L \left(\frac{\rho(1+\nu)(1-2\nu)}{E(1-\nu)} \right)^{1/2} \tag{10.50}$$

where L is the smallest length between any two nodes and μ is a coefficient dependent on the type of element employed.[5] For problems in which many time steps are used it may be beneficial to calculate the exact highest linear frequency of the finite element mesh prior to the time stepping analysis.

Recall that when an elasto-viscoplastic model is adopted care must be taken not to exceed the critical time step for the Euler scheme in evaluating the viscoplastic strains. (See Section 8.3).

10.6 Program DYNPAK

10.6.1 Overall structure of DYNPAK

We now present program DYNPAK for the elasto-viscoplastic or geometrically nonlinear, transient dynamic analysis of plane stress, plane strain and axisymmetric problems. The basic structure of the program is shown in Fig. 10.2. Many of the subroutines used in DYNPAK have already been described in earlier chapters.

The algorithm adopted has been presented schematically in Fig. 10.1. The program is written in a dynamically dimensioned form. Efficiency has sometimes been sacrificed for clarity of presentation and the reader may consider ways of making the program more efficient when reviewing this chapter.

Isoparametric 4, 8 and 9-noded quadrilateral elements are included in the program. A special mass lumping procedure[6] has been adopted and separate Gauss–Legendre rules may be adopted in the evaluation of the stiffness and the lumped mass matrices.

Impact and seismic loading can easily be specified. Material nonlinearity is based on elasto-viscoplastic models with Von Mises, Tresca, Mohr–Coulomb or Drucker–Prager yield criteria with isotropic hardening. A total Lagrangian formulation is used to allow for the geometric nonlinear behaviour.

Subroutines GAUSSQ, SFR2 and JACOB2 have already been dealt with and only the remaining routines will be listed and described.

10.6.2 Master routine DYNPAK

The master routine organises the calling of the main routines as outlined in Fig. 10.2. In subroutine CONTOL the variables required for dynamic dimensioning are read and a check is made on the maximum available dimensions. Note that the values given in the DIMENSION statement in

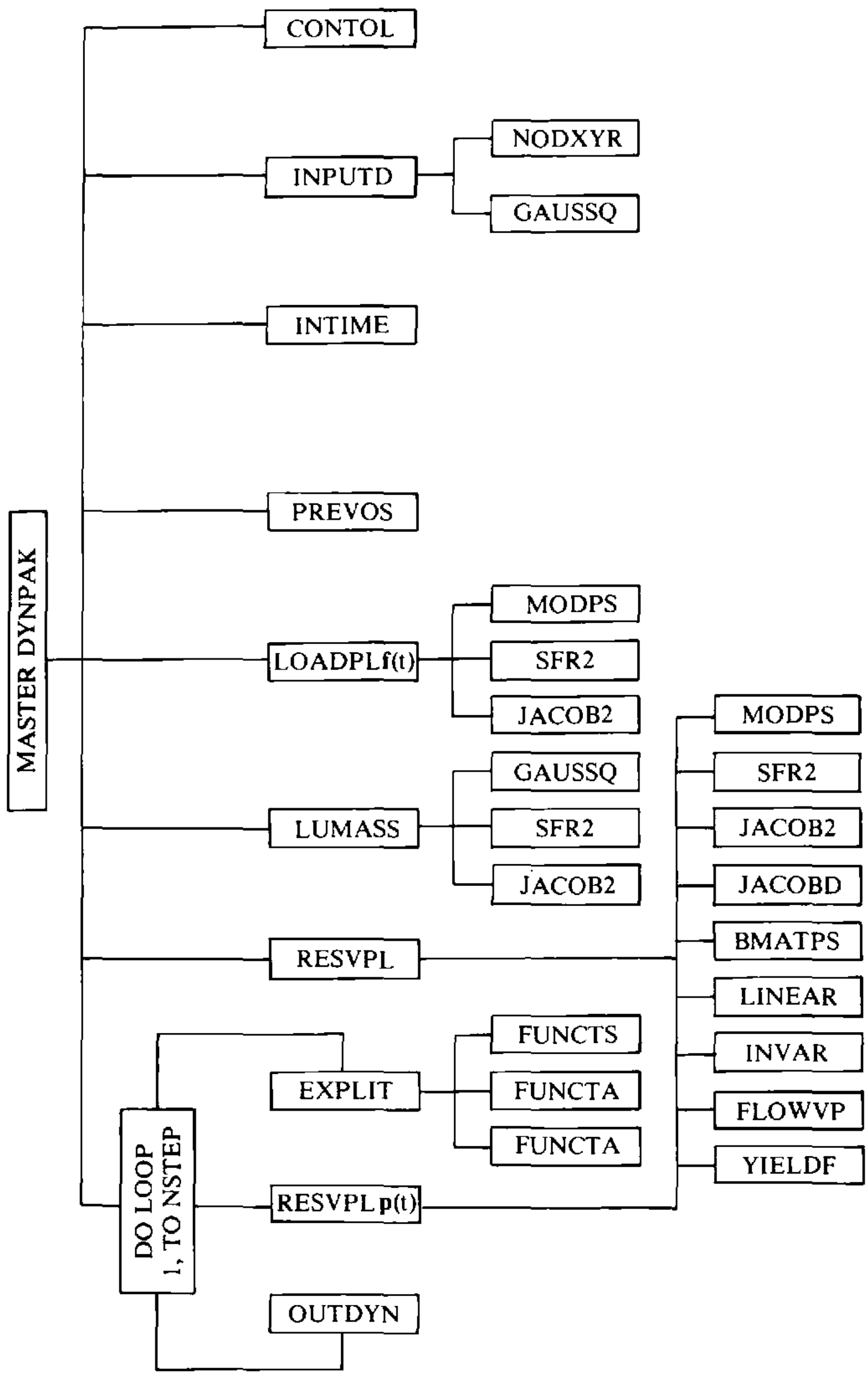


Fig. 10.2 Flow diagram for program DYNPAK.

DYNPAK should agree with the values specified in CONTOL. Subroutines INPUTD, INTIME and PREVOS read the mesh data, the time integration data and data for the previous state of the structure. Subroutines LUMASS and LOADPL generate the lumped mass and applied force vectors respectively. FIXITY deals with fixed boundary nodes. In the time step do loop, EXPLIT performs the direct time integration and RESVPL calculates

$$\int_{\Omega} [\boldsymbol{B}]_n{}^T \boldsymbol{\sigma}_n \, d\Omega$$

when an elasto-viscoplastic material model is adopted.

In this version of DYNPAK it should be noted that the maximum dimensions imply that we can solve problems with no more than 50 elements, 200 nodal points, 50 fixed boundary nodes and 600 acceleration ordinates.

Of course, larger problems can be accommodated by increasing the values in CONTOL and also the appropriate dimensions in the DIMENSION statement in the main routine DYNPAK.

```
      PROGRAM DYNPAK (INPUT ,TAPE5=INPUT ,TAPE4,TAPE10,TAPE12,TAPE3,
     .                OUTPUT,TAPE6=OUTPUT,TAPE7,TAPE11,TAPE13)
C**********************************************************************
C
C     DYNAMIC TRANSIENT ELASTO - VISCOPLASTIC PROGRAM
C
C**********************************************************************
      DIMENSION ACCEH(  600) ,ACCEV(  600) ,COORD(200,2) ,DISPL(  400) ,
     .          FORCE(  400) ,IFPRE(2,200) ,LNODS(50 ,9) ,MATNO(   50) ,
     .          INTGR(   50) ,NPRQD(   10) ,NGRQS(   10) ,POSGP(    4) ,
     .          PROPS(10,13) ,RESID(  400) ,RLOAD(50,18) ,STRIN(4,450) ,
     .          STRSG(4,450) ,TDISP(  400) ,TEMPE(  100) ,VELOC(  400) ,
     .          VISTN(4,450) ,VIVEL(5,450) ,WEIGP(    4) ,YMASS(  400)
C
      CALL      CONTOL     (NDOFN ,NELEM ,NMATS ,NPOIN )
C
      CALL      INPUTD     (COORD ,IFPRE ,LNODS ,MATNO ,NCONM ,NCRIT ,
     .                      NDIME ,NDOFN ,NELEM ,NGAUM ,NGAUS ,NLAPS ,
     .                      NMATS ,NNODE ,NPOIN ,NPREV ,NSTRE ,NTYPE ,
     .                      POSGP ,PROPS ,WEIGP )
C
      CALL      INTIME     (AALFA ,ACCEH ,ACCEV ,AFACT ,AZERO ,BEETA ,
     .                      BZERO ,DELTA ,DTIME ,DTEND ,GAAMA ,IFIXD ,
     .                      IFUNC ,INTGR ,KSTEP ,MITER ,NDOFN ,NELEM ,
     .                      NGRQS ,NOUTD ,NOUTP ,NPOIN ,NPRQD ,NREQD ,
     .                      NREQS ,NSTEP ,OMEGA ,TDISP ,TOLER ,VELOC ,
     .                      IPRED )
C
      CALL      PREVOS     (FORCE ,NDOFN ,NELEM ,NGAUS ,NPOIN ,NPREV ,
     .                      STRIN )
C
      CALL      LOADPL     (COORD ,FORCE ,LNODS ,MATNO ,NDIME ,NDOFN ,
     .                      NELEM ,NGAUS ,NMATS ,NNODE ,NPOIN ,NSTRE ,
     .                      NTYPE ,POSGP ,PROPS ,RLOAD ,STRIN ,TEMPE ,
     .                      WEIGP )
C
      CALL      LUMASS     (COORD ,INTGR ,LNODS ,MATNO ,NCONM ,NDIME ,
     .                      NDOFN ,NELEM ,NGAUM ,NMATS ,NNODE ,NPOIN ,
     .                      NTYPE ,PROPS ,YMASS )
```

```
C
      CALL        FIXITY    (IFPRE ,NDOFN ,NPOIN ,YMASS )
C
      IF(NPREV.NE.0)
     .CALL        RESVPL    (COORD ,DTIME ,LNODS ,MATNO ,NCRIT, NDIME ,
     .                       NDOFN ,NELEM ,NGAUS ,NLAPS ,NNODE ,NMATS ,
     .                       NPOIN ,NSTRE ,NTYPE ,POSGP ,PROPS ,RESID ,
     .                       RLOAD ,STRIN ,STRSG ,TDISP ,VISTN ,VIVEL ,
     .                       WEIGP )
C
      DO 500 ISTEP=1,NSTEP
C
      CALL        EXPLIT    (ACCEH ,ACCEV ,AFACT ,AZERO ,AALFA ,BZERO ,
     .                       DTIME ,DTEND ,FORCE ,IFIXD ,IFPRE ,IFUNC ,
     .                       ISTEP ,NDOFN ,NPOIN ,OMEGA ,RESID ,TDISP ,
     .                       VELOC ,YMASS )
C
      CALL        RESVPL    (COORD ,DTIME ,LNODS ,MATNO ,NCRIT ,NDIME ,
     .                       NDOFN ,NELEM ,NGAUS ,NLAPS ,NNODE ,NMATS ,
     .                       NPOIN ,NSTRE ,NTYPE ,POSGP ,PROPS ,RESID ,
     .                       RLOAD ,STRIN ,STRSG ,TDISP ,VISTN ,VIVEL ,
     .                       WEIGP )
C
      CALL        OUTDYN    (DISPL ,DTIME ,ISTEP ,NDOFN ,NELEM ,NGAUS ,
     .                       NGRQS ,NOUTD ,NOUTP ,NPOIN ,NPRQD ,NREQD ,
     .                       NREQS ,NTYPE ,STRSG ,TDISP ,VIVEL )
C
  500 CONTINUE
      STOP
      END
```

10.6.3 Subroutine BLARGE

This subroutine evaluates the strain-displacement matrix for geometrically nonlinear displacements using the deformation Jacobian matrix $[J_D]_n$. Note that for small displacement analysis we pre-set NLAPS = 0.

```
      SUBROUTINE BLARGE    (BMATX ,CARTD ,DJACM ,DLCOD ,GPCOD ,KGASP ,
     .                      NLAPS ,NNODE ,NTYPE ,SHAPE )
C*********************************************************************
C
C*** LARGE DISPLACEMENT B MATRIX
C
C*********************************************************************
      DIMENSION BMATX(4,18),CARTD(2,9),DJACM(2,2),DLCOD(2,9),
     .          GPCOD(2, 9),SHAPE(  9)
      NGASH=0
      DO 10 INODE=1,NNODE
      MGASH=NGASH+1
      NGASH=MGASH+1
      BMATX(1,MGASH)=CARTD(1,INODE)*DJACM(1,1)
      BMATX(1,NGASH)=CARTD(1,INODE)*DJACM(2,1)
      BMATX(2,MGASH)=CARTD(2,INODE)*DJACM(1,2)
      BMATX(2,NGASH)=CARTD(2,INODE)*DJACM(2,2)
      BMATX(3,MGASH)=CARTD(2,INODE)*DJACM(1,1)+CARTD(1,INODE)*DJACM(1,2)
      BMATX(3,NGASH)=CARTD(1,INODE)*DJACM(2,2)+CARTD(2,INODE)*DJACM(2,1)
   10 CONTINUE
      IF(NTYPE.NE.3) RETURN
      FMULT=1.
      IF(NLAPS.LT.2) GO TO 40
      FMULT=0.0
```

```
      DO 20 JNODE=1,NNODE                                                BLAR 25
   20 FMULT=FMULT+DLCOD(1,JNODE)*SHAPE(JNODE)                            BLAR 26
      FMULT=FMULT/GPCOD(1,KGASP)                                         BLAR 27
   40 NGASH=0                                                            BLAR 28
      DO 30 INODE=1,NNODE                                                BLAR 29
      MGASH=NGASH+1                                                      BLAR 30
      NGASH=MGASH+1                                                      BLAR 31
      BMATX(4,MGASH)=SHAPE(INODE)*FMULT/GPCOD(1,KGASP)                   BLAR 32
   30 BMATX(4,NGASH)=0.0                                                 BLAR 33
      RETURN                                                             BLAR 34
      END                                                                BLAR 35
```

BLAR 10–20 Evaluate the complete strain matrix for plane stress/strain problems and the first three rows of the strain matrix for axisymmetric problems.

BLAR 21–33 Evaluate the remainder of the strain matrix for axisymmetric problems, if applicable.

10.6.4 Subroutine CONTOL

The purpose of this subroutine is to set the values of variables for the dynamic dimensions which are used elsewhere in the program. If any change in the DIMENSION statement in the master routine is made, then a corresponding change in this subroutine should also be made.

```
      SUBROUTINE CONTOL (NDOFN ,NELEM ,NMATS ,NPOIN )
C*********************************************************************
C
C*** READ CONTROL DATA AND CHECK FOR DIMENSIONS
C
C*********************************************************************
      READ(5,110) NPOIN,NELEM,NDOFN,NMATS
      IF(NELEM.GT. 50) GO TO 200
      IF(NPOIN.GT.200) GO TO 200
      IF(NMATS.GT. 10) GO TO 200
      GO TO 210
  200 WRITE(6,120)
      STOP
  120 FORMAT(/'SET DIMENSION EXCEEDED - CONTOL CHECK '/)
  110 FORMAT(16I5)
  210 CONTINUE
      RETURN
      END
```

10.6.5 Subroutine EXPLIT

This subroutine performs the direct time integration using expressions (10.43) and (10.44) to evaluate the nodal displacements at every time step. Special provisions are made for the first time step.

```
      SUBROUTINE EXPLIT (ACCEH ,ACCEV ,AFACT ,AZERO ,AALFA ,BZERO ,
     .                   DTIME ,DTEND ,FORCE ,IFIXD ,IFPRE ,IFUNC ,
     .                   ISTEP ,NDOFN ,NPOIN ,OMEGA ,RESID ,TDISP ,
     .                   VELOC ,YMASS )
C*********************************************************************
C
C *** TIME STEPPING ROUTINE
C
C*********************************************************************
```

```
      DIMENSION YMASS(1),ACCEH(1),TDISP(1),RESID(  1),                 EXPL  10
                FORCE(1),ACCEV(1),VELOC(1),IFPRE(2,1)                  EXPL  11
      CFACT=1.0+0.5*AALFA*DTIME                                        EXPL  12
      CFACT=1./CFACT                                                   EXPL  13
      CONS1=2.*CFACT                                                   EXPL  14
      RCONS=1./CONS1                                                   EXPL  15
      CONS2=CONS1-1                                                    EXPL  16
      CONS3=DTIME*DTIME*CFACT                                          EXPL  17
      CONS4=-2.0*CONS2*DTIME                                           EXPL  18
      IF(ISTEP.GT.1) CONS4=CONS2                                       EXPL  19
      NPOSN=0                                                          EXPL  20
      FACTS=FUNCTS (AZERO,BZERO,DTEND,DTIME,IFUNC,ISTEP,OMEGA)         EXPL  21
      FACTH=FUNCTA (ACCEH,AFACT,DTEND,DTIME,IFUNC,ISTEP)               EXPL  22
      FACTV=FUNCTA (ACCEV,AFACT,DTEND,DTIME,IFUNC,ISTEP)               EXPL  23
      DO 500 IPOIN=1,NPOIN                                             EXPL  24
      DO 510 IDOFN=1,NDOFN                                             EXPL  25
      FACTT=0.0                                                        EXPL  26
      IF(IFUNC.NE.0) GO TO 200                                         EXPL  27
      IF(IFIXD.EQ.0.AND.IDOFN.EQ.1) FACTT=FACTH                        EXPL  28
      IF(IFIXD.EQ.0.AND.IDOFN.EQ.2) FACTT=FACTV                        EXPL  29
      IF(IFIXD.EQ.1.AND.IDOFN.EQ.2) FACTT=FACTV                        EXPL  30
      IF(IFIXD.EQ.2.AND.IDOFN.EQ.1) FACTT=FACTH                        EXPL  31
      IF(IFPRE(IDOFN,IPOIN).EQ.0) GO TO 200                            EXPL  32
      FACTT=0.0                                                        EXPL  33
      FACTS=1.0                                                        EXPL  34
  200 CONTINUE                                                         EXPL  35
      NPOSN=NPOSN+1                                                    EXPL  36
      DCURR=TDISP(NPOSN)                                               EXPL  37
      RESID(NPOSN)=RESID(NPOSN)-FORCE(NPOSN)*FACTS                     EXPL  38
      TDISP(NPOSN)=-RESID(NPOSN)*CONS3/YMASS(NPOSN)                    EXPL  39
     .-FACTT*CONS3+DCURR*CONS1-VELOC(NPOSN)*CONS4                      EXPL  40
      IF(ISTEP.EQ.1) TDISP(NPOSN)=TDISP(NPOSN)*RCONS                   EXPL  41
      VELOC(NPOSN)=DCURR                                               EXPL  42
  510 CONTINUE                                                         EXPL  43
  500 CONTINUE                                                         EXPL  44
      RETURN                                                           EXPL  45
      END                                                              EXPL  46
```

EXPL 12–19 Evaluate the various time integration constants. After the first time step modify variable CONS4.

EXPL 21 Evaluate the value of the time varying Heaviside or harmonic function for a particular time step.

EXPL 22–23 Evaluate the acceleration ordinates (FACTH for horizontal and FACTV for vertical acceleration respectively) at a particular time step.

EXPL 24–31 The seismic force is only applied for particular degrees of freedom. For IFIXD = 1 only vertical, IFIXD = 2 only horizontal or radial and IFIXD = 0 both components of the acceleration are considered.

EXPL 32–35 Assign appropriate values for restrained boundary nodes.

EXPL 36–40 Evaluate displacements.

EXPL 41 For the first time step modify the displacement.

EXPL 42 Store the current displacements for the next time step.

10.6.6 Subroutine FIXITY

This subroutine deals with the restrained degrees of freedom (boundary points). The diagonal mass vector, XMASS, is modified—for restrained degrees of freedom. The component of the XMASS vector is set to a large value such as 1.E30, which artificially makes the displacement zero.

```
      SUBROUTINE FIXITY    (IFPRE ,NDOFN ,NPOIN ,YMASS )
C*********************************************************************
C
C *** DEALS WITH FIXED BOUNDARY NODES
C
C*********************************************************************
      DIMENSION IFPRE(2,1) ,YMASS(1)
      NTOTV=NDOFN*NPOIN
      IPOSN=0
      DO 500 IPOIN=1,NPOIN
      DO 500 IDOFN=1,NDOFN
      IPOSN=IPOSN+1
  500 IF(IFPRE(IDOFN,IPOIN).EQ.1) YMASS(IPOSN)= 1.E30
      WRITE(6,900)
  900 FORMAT(/5X,19HNODAL LUMPED MASSES/)
      WRITE(6,910) (ITOTV,YMASS(ITOTV),ITOTV=1,NTOTV)
  910 FORMAT(6(1X,I5,E13.5))
      RETURN
      END
```

10.6.7 Subroutine FLOWVP

This routine evaluates the viscoplastic strain rate.

```
      SUBROUTINE FLOWVP (AVECT ,KGAUS ,LPROP ,NCRIT ,NMATS ,PROPS ,
     .                   STEFF ,VIVEL ,YIELD )
C*********************************************************************
C
C **** CALCULATES VISCOPLASTIC STRAIN RATE
C
C*********************************************************************
      DIMENSION AVECT(4) ,PROPS(NMATS,1) ,VIVEL(5,1)
      IF(STEFF.EQ.0.0) GO TO 90
      NSTR1=4
      TOLOR=0.01
      FDATM=PROPS(LPROP, 6)
      HARDS=PROPS(LPROP, 7)
      FRICT=PROPS(LPROP, 8)
      GAMMA=PROPS(LPROP, 9)
      DELTA=PROPS(LPROP,10)
      NFLOW=PROPS(LPROP,11)
      FRICT=FRICT*0.017453292
      IF(NCRIT.EQ.3) FDATM=FDATM*COS(FRICT)
      IF(NCRIT.EQ.4) FDATM=6.0*FDATM*COS(FRICT)/
     .(1.73205080757*(3.0-SIN(FRICT)))
      IF(HARDS.GT.0.) FDATM=FDATM+VIVEL(5,KGAUS)*HARDS
      IF(FDATM.LT.0.001) FDATM=1.0
      FCURR=YIELD-FDATM
      FNORM=FCURR/FDATM
      IF(FNORM.LT.TOLOR) GO TO 90
      IF(NFLOW.EQ.1) GO TO 50
      CMULT=GAMMA*(EXP(DELTA*FNORM)-1.0)
      GO TO 60
   50 CMULT=GAMMA*(FNORM**DELTA)
   60 CONTINUE
      DO 70 ISTR1=1,NSTR1
   70 AVECT(ISTR1)=CMULT*AVECT(ISTR1)
      DO 80 ISTR1=1,NSTR1
   80 VIVEL(ISTR1,KGAUS)=AVECT(ISTR1)
      RETURN
   90 DO 100 ISTR1=1,NSTR1
  100 VIVEL(ISTR1,KGAUS)=0.
      RETURN
      END
```

10.6.8 Function FUNCTA

This function interpolates the accelerogram data for a particular time step. AFACT is the ratio of the accelerogram record time step length to the computational time step length.

```
      FUNCTION FUNCTA  (ACCER,AFACT,DTEND,DTIME,IFUNC,JSTEP)
C***********************************************************************
C
C***  ACCELEROGRAM INTERPOLATION
C
C***********************************************************************
      DIMENSION ACCER(1)
      IF(IFUNC.NE.0) RETURN
      FUNCTA=0.0
      IF(JSTEP.EQ.0.OR.FLOAT(JSTEP)*DTIME.GT.DTEND) RETURN
      XGASH=(FLOAT(JSTEP)-1.0)/AFACT+1.0
      MGASH=XGASH
      NGASH=MGASH+1
      XGASH=XGASH-FLOAT(MGASH)
      FUNCTA=ACCER(MGASH)*(1.0-XGASH)+XGASH*ACCER(NGASH)
      RETURN
      END
```

10.6.9 Function FUNCTS

This function sets the value of the time varying function for a particular time step. Heaviside functions ($f(t) = 1.0\ H(t)$) or harmonic functions, ($f(t) = a - b \sin \omega t$) can be specified.

```
      FUNCTION FUNCTS  (AZERO,BZERO,DTEND,DTIME,IFUNC,JSTEP,OMEGA)
C***********************************************************************
C
C***  HEAVISIDE AND HARMONIC TIME FUNCTION
C
C***********************************************************************
      IF(IFUNC.EQ.0) RETURN
      FUNCTS=0.0
      IF(JSTEP.EQ.0.OR.FLOAT(JSTEP)*DTIME.GT.DTEND) RETURN
      IF(IFUNC.EQ.1) FUNCTS = 1.0
      IF(IFUNC.EQ.2) ARGUM=OMEGA*JSTEP*DTIME
      IF(IFUNC.EQ.2) FUNCTS = AZERO + BZERO*SIN(ARGUM)
      RETURN
      END
```

10.6.10 Subroutine INPUTD

This subroutine reads and writes most of the control parameters, nodal point coordinates, element connectivities, boundary conditions and material properties. It also writes the geometric data onto file 13 for deformation plotting. A similar routine was described in Chapter 6.

```
      SUBROUTINE INPUTD    (COORD ,IFPRE ,LNODS ,MATNO ,NCONM ,NCRIT ,
     .                      NDIME ,NDOFN ,NELEM ,NGAUM ,NGAUS ,NLAPS ,
     .                      NMATS ,NNODE ,NPOIN ,NPREV ,NSTRE ,NTYPE ,
     .                      POSGP ,PROPS ,WEIGP )
C***********************************************************************
C
C***  DYNPAK INPUT ROUTINE
C
C***********************************************************************
      DIMENSION COORD(NPOIN,1) ,IFPRE(NDOFN,1) ,WEIGP(1) ,MATNO(1) ,
```

```
      READ(5,913) TITLE
  913 FORMAT(10A4)
      WRITE(6,914) TITLE
  914 FORMAT(//,5X,10A4)
C
C*** READ THE FIRST DATA CARD, AND ECHO IT IMMEDIATELY.
C
      READ (5,900) NVFIX,NTYPE,NNODE,NPROP,NGAUS,NDIME,NSTRE,NCRIT,
     .             NPREV,NCONM,NLAPS,NGAUM,NRADS
      WRITE(6,901) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NDOFN,NMATS,NPROP,
     .             NGAUS,NDIME,NSTRE,NCRIT,NPREV,NCONM,NLAPS,NGAUM,
     .             NRADS
  901 FORMAT (/5X,18HCONTROL PARAMETERS/
     .         /5X,8H NPOIN =,I10,5X,8H NELEM =,I10,5X,8H NVFIX =,I10/
     .         /5X,8H NTYPE =,I10,5X,8H NNODE =,I10,5X,8H NDOFN =,I10/
     .         /5X,8H NMATS =,I10,5X,8H NPROP =,I10,5X,8H NGAUS =,I10/
     .         /5X,8H NDIME =,I10,5X,8H NSTRE =,I10,5X,8H NCRIT =,I10/
     .         /5X,8H NPREV =,I10,5X,8H NCONM =,I10,5X,8H NLAPS =,I10/
     .         /5X,8H NGAUM =,I10,5X,8H NRADS =,I10/)
  900 FORMAT(16I5)
C
C *** READ THE ELEMENT NODAL CONNECTIONS, AND THE PROPERTY NUMBERS.
C
      WRITE (6,902)
  902 FORMAT(//5X,8H ELEMENT,3X,8HPROPERTY,6X,12HNODE NUMBERS)
      DO 530 IELEM=1,NELEM
      READ (5,900) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
      WRITE(13,915 ) NUMEL,(LNODS(NUMEL,INODE),INODE=1,NNODE)
  530 WRITE(6,903) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
  903 FORMAT(6X,I5,I9,6X,10I5)
  915 FORMAT(16I5)
C
C*** ZERO ALL THE NODAL COORDINATES, PRIOR TO READING SOME OF THEM.
C
      DO 500 IPOIN=1,NPOIN
      DO 500 IDIME=1,NDIME
  500 COORD(IPOIN,IDIME)=0.0
C
C*** READ SOME NODAL COORDINATES, FINISHING WITH THE LAST NODE OF ALL.
C
  904 FORMAT(//5X,5H NODE,9X,1HX,9X,1HY,5X)
  200 READ (5,905) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,NDIME)
      WRITE(6,906) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,NDIME)
  905 FORMAT(I5,6F10.5)
      IF (IPOIN.NE.NPOIN) GO TO 200
C
C*** INTERPOLATE COORDINATES OF MID-SIDE NODES
C
      CALL NODXYR (COORD,LNODS,NELEM,NNODE,NPOIN,NRADS,NTYPE)
C
      WRITE (6,904)
      WRITE(13,916 ) (IPOIN,(COORD(IPOIN,IDIME),IDIME=1,NDIME),
     .  IPOIN=1,NPOIN)
  916 FORMAT(I5,2G15.6)
      WRITE( 6, 906) (IPOIN,(COORD(IPOIN,IDIME),IDIME=1,NDIME),
     .IPOIN=1,NPOIN)
  906 FORMAT(5X,I5,2F10.3)
C
C*** READ THE FIXED VALUES.
C
      WRITE(6,907)
  907 FORMAT(//5X,5H NODE,2X,4HCODE)
      DO 540 IPOIN=1,NPOIN
      DO 540 IDOFN=1,NDOFN
```

```
  540 IFPRE(IDOFN,IPOIN)=0
      DO 550 IVFIX=1,NVFIX
  550 READ (5,908) IPOIN,(IFPRE(IDOFN,IPOIN),IDOFN=1,NDOFN)
      DO 560 IPOIN=1,NPOIN
  560 WRITE(6,909) IPOIN,(IFPRE(IDOFN,IPOIN),IDOFN=1,NDOFN)
  908 FORMAT(1X,I4,3X,2I1)
  909 FORMAT(6X,I5,3X,2I1)
C
C*** READ THE AVAILABLE SELECTION OF ELEMENT PROPERTIES.
C
      WRITE(6,910)
  910 FORMAT(//5X,19HMATERIAL PROPERTIES)
      DO 520 IMATS=1,NMATS
      READ(5,900) NUMAT
      READ (5,917) (PROPS(NUMAT,IPROP),IPROP=1,NPROP)
      WRITE(6,911) NUMAT
  911 FORMAT(/5X,11HMATERIAL NO,I5)
  520 WRITE(6,912) (PROPS(NUMAT,IPROP),IPROP=1,NPROP)
  912 FORMAT(/5X,13HYOUNG MODULUS,G12.4/5X,13HPOISSON RATIO,G12.4/
     .        5X,13HTHICKNESS    ,G12.4/5X,13HMASS DENSITY ,G12.4/
     .        5X,13HALPHA TEMPR  ,G12.4/5X,13HREFERENCE FO ,G12.4/
     .        5X,13HHARDENING PAR,G12.4/5X,13HFRICT ANGLE  ,G12.4/
     .        5X,13HFLUIDITY PAR ,G12.4/5X,13HEXP DELTA    ,G12.4/
     .        5X,13HNFLOW CODE   ,G12.4)
  917 FORMAT(8E10.4)
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL GAUSSQ (NGAUS,POSGP,WEIGP)
      RETURN
      END
```

10.6.11 Subroutine INTIME

This routine reads and writes all data required for time integration and plotting stress and displacement histories.

```
      SUBROUTINE INTIME (AALFA ,ACCEH ,ACCEV ,AFACT ,AZERO ,BEETA ,
     .                   BZERO ,DELTA ,DTIME ,DTEND ,GAAMA ,IFIXD ,
     .                   IFUNC ,INTGR ,KSTEP ,MITER ,NDOFN ,NELEM ,
     .                   NGRQS ,NOUTD ,NOUTP ,NPOIN ,NPRQD ,NREQD ,
     .                   NREQS ,NSTEP ,OMEGA ,TDISP ,TOLER ,VELOC ,
     .                   IPRED )
C***********************************************************************
C
C ** INITIAL VALUES AND TIME INTEGRATION DATA
C
C***********************************************************************
      DIMENSION TDISP(1) ,ACCEH(1) ,NPRQD(1) ,INTGR(1) ,
     .          VELOC(1) ,ACCEV(1) ,NGRQS(1)
C
C***  READ TIME STEPPING AND SELECTIVE OUTPUT PARAMETERS
C
      READ (5,902) NSTEP,NOUTD,NOUTP,NREQD,NREQS,NACCE,IFUNC,
     . .           IFIXD,MITER,KSTEP,IPRED
      READ (5,190) DTIME,DTEND,DTREC,AALFA,BEETA,DELTA,GAAMA,
     .             AZERO,BZERO,OMEGA,TOLER
      WRITE(6,950) NSTEP,NOUTD,NOUTP,NREQD,NREQS,NACCE,IFUNC,
     .             IFIXD,MITER,KSTEP,IPRED
      WRITE(6,960) DTIME,DTEND,DTREC,AALFA,BEETA,DELTA,GAAMA,
     .             AZERO,BZERO,OMEGA,TOLER
  950 FORMAT(/5X,'TIME STEPPING PARAMETERS'/
     .       /5X,'NSTEP=',I5,12X,'NOUTD=',I5,12X,'NOUTP=',I5,/
     .       /5X,'NREQD=',I5,12X,'NREQS=',I5,12X,'NACCE=',I5,/
     .       /5X,'IFUNC=',I5,12X,'IFIXD=',I5,12X,'MITER=',I5,/
     .       /5X,'KSTEP=',I5,12X,'IPRED=',I5)
```

```
  960 FORMAT(/5X,'DTIME=',G12.4,5X,'DTEND=',G12.4,5X,'DTREC=',G12.4,/
     .       /5X,'AALFA=',G12.4,5X,'BEETA=',G12.4,5X,'DELTA=',G12.4,/
     .       /5X,'GAAMA=',G12.4,5X,'AZERO=',G12.4,5X,'BZERO=',G12.4,/
     .       /5X,'OMEGA=',G12.4,5X,'TOLER=',G12.4)
C
C***  SELECTED NODES AND GAUSS POINTS FOR OUTPUT
C
      READ(5,902) (NPRQD(IREQD),IREQD=1,NREQD)
      READ(5,902) (NGRQS(IREQS),IREQS=1,NREQS)
      WRITE(6,909)
  909 FORMAT(//5X,41H SELECTIVE OUTPUT REQUESTED FOR FOLLOWING )
      WRITE(6,910) (NPRQD(IREQD),IREQD=1,NREQD)
  910 FORMAT(/,5X,6H NODES,10I5)
      WRITE(6,911) (NGRQS(IREQS),IREQS=1,NREQS)
  911 FORMAT(5X,6H G.P. ,10I5)
  902 FORMAT(16I5)
  190 FORMAT(8F10.4)
C
C***   READ THE INDICATOR FOR EXPLICIT OR IMPLICIT ELEMENT
C
      READ (5,902) (INTGR(IELEM),IELEM=1,NELEM)
      WRITE(6,930)
      WRITE(6,902) (INTGR(IELEM),IELEM=1,NELEM)
  930 FORMAT(/5X, '  TYPE OF ELEMENT, IMPLICIT=1,EXPLICIT=2 '/)
C
C*** INITIAL DISPLACEMENTS
C
      JPOIN=0
      DO 500 IPOIN=1,NPOIN
      DO 500 IDOFN=1,NDOFN
      JPOIN=JPOIN+1
      TDISP(JPOIN)=0.
  500 VELOC(JPOIN)=0.
      WRITE(6,903)
  200 READ(5,904) NGASH,XGASH,YGASH
      NPOSN=(NGASH-1)*NDOFN+1
      TDISP(NPOSN)=XGASH
      NPOSN=NPOSN+1
      TDISP(NPOSN)=YGASH
      WRITE(6,905) NGASH,XGASH,YGASH
      IF(NGASH.NE.NPOIN) GO TO 200
C
C*** INITIAL VELOCITIES
C
      WRITE(6,906)
  210 READ(5,904) NGASH,XGASH,YGASH
      NPOSN=(NGASH-1)*NDOFN+1
      VELOC(NPOSN)=XGASH
      NPOSN=NPOSN+1
      VELOC(NPOSN)=YGASH
      WRITE(6,905) NGASH,XGASH,YGASH
      IF(NGASH.NE.NPOIN) GO TO 210
  904 FORMAT(I5,2F10.5)
  903 FORMAT(//5X,5H NODE,2X,16H INITIAL X-DISP.,2X,
     .16H INITIAL Y-DISP./)
  905 FORMAT(I10,2E18.5)
  906 FORMAT(//5X,5H NODE,2X,16H INITIAL X-VELO.,2X,
     . 16H INITIAL Y-VELO./)
      IF (IFUNC.NE.0) GO TO 250
C
C*** READ ACCELEROGRAM DATA ,X-DIREC FROM TAPE 7,Y-DIREC FROM TAPE 12
C
      AFACT=DTREC/DTIME
      IF(IFIXD-1) 220,230,240
  220 READ (7,907)(ACCEH(I),I=1,NACCE)
```

```
      WRITE(6,912) DTREC                                                TIME  95
      WRITE(6,907)(ACCEH(I),I=1,NACCE)                                  TIME  96
      READ(12,907)(ACCEV(I),I=1,NACCE)                                  TIME  97
      WRITE(6,913) DTREC                                                TIME  98
      WRITE(6,907)(ACCEV(I),I=1,NACCE)                                  TIME  99
      GO TO 250                                                         TIME 100
  230 READ(12,907)(ACCEV(I),I=1,NACCE)                                  TIME 101
      WRITE(6,913) DTREC                                                TIME 102
      WRITE(6,907)(ACCEV(I),I=1,NACCE)                                  TIME 103
      GO TO 250                                                         TIME 104
  240 READ(7,907) (ACCEH(I),I=1,NACCE)                                  TIME 105
      WRITE(6,912)                                                      TIME 106
      WRITE(6,907)(ACCEH(I),I=1,NACCE)                                  TIME 107
  907 FORMAT(7F10.3)                                                    TIME 108
  912 FORMAT(/5X,'HORIZONTAL ACCELERATION ORDINATES AT',F9.4,2X,'SEC'/) TIME 109
  913 FORMAT(/5X,'VERTICAL   ACCELERATION ORDINATES AT',F9.4,2X,'SEC'/) TIME 110
  250 CONTINUE                                                          TIME 111
      RETURN                                                            TIME 112
      END                                                               TIME 113
```

TIME 14–33	Read and write most of the control time integration data.
TIME 34–46	Read the selective nodal points and integration points for displacement and stress history.
TIME 54–70	Read initial displacement.
TIME 71–87	Read initial velocities.
TIME 89–111	Read appropriate acceleration data.

10.6.12 Subroutine INVAR

This routine calculates the stress invariants and yield values for the various yield criteria. The choice of yield criterion is governed by the parameter NCRIT. A similar routine was described in Section 7.8.3.

```
      SUBROUTINE INVAR    (DEVIA ,LPROP ,NCRIT ,NMATS ,PROPS ,SINT3 ,
     .                     STEFF ,STEMP ,THETA ,VARJ2 ,YIELD )
C***********************************************************************
C
C** STRESS INVARIANTS
C
C***********************************************************************
      DIMENSION DEVIA(4) ,PROPS(NMATS,1) ,STEMP(4)
C
C***  INVARIANTS
C
      ROOT3=1.73205080757
      SMEAN=(STEMP(1)+STEMP(2)+STEMP(4))/3.0
      DEVIA(1)=STEMP(1)-SMEAN
      DEVIA(2)=STEMP(2)-SMEAN
      DEVIA(3)=STEMP(3)
      DEVIA(4)=STEMP(4)-SMEAN
      VARJ2=DEVIA(3)*DEVIA(3)+0.5*(DEVIA(1)*DEVIA(1)+
     . DEVIA(2)*DEVIA(2)+DEVIA(4)*DEVIA(4))
      VARJ3=DEVIA(4)*(DEVIA(4)*DEVIA(4)-VARJ2)
      STEFF=SQRT(VARJ2)
      IF (VARJ2.EQ.0.0.OR.STEFF.EQ.0.0) GO TO 5
      SINT3=-2.5980762113*VARJ3/(VARJ2*STEFF)
      GO TO 6
    5 SINT3=0.0
    6 CONTINUE
      IF(SINT3.LT.-1.0) SINT3=-1.0
```

```
      IF(SINT3.GT. 1.0) SINT3= 1.0
      THETA=ASIN(SINT3)/3.0
      GO TO (1,2,3,4) NCRIT
C*** TRESCA
    1 YIELD=2.0*COS(THETA)*STEFF
      RETURN
C*** VON MISES
    2 YIELD=ROOT3*STEFF
      RETURN
C*** MOHR-COULOMB
    3 PHIRA=PROPS(LPROP,8)*0.017453292
      SNPHI=SIN(PHIRA)
      YIELD=SMEAN*SNPHI+STEFF*(COS(THETA)-SIN(THETA)*SNPHI/ROOT3)
      RETURN
C*** DRUCKER-PRAGER
    4 PHIRA=PROPS(LPROP,8)*0.017453292
      SNPHI=SIN(PHIRA)
      YIELD=6.0*SMEAN*SNPHI/(ROOT3*(3.0-SNPHI))+STEFF
      RETURN
      END
```

10.6.13 Subroutine JACOBD

This subroutine evaluates the deformation Jacobian matrix $[J_D]_n$ for a particular sampling point within an element.

```
      SUBROUTINE JACOBD   (CARTD ,DLCOD ,DJACM ,NDIME ,NLAPS ,NNODE )
C***********************************************************************
C
C*** DEFORMATION JACOBIAN
C
C***********************************************************************
      DIMENSION CARTD(2,9) ,DLCOD(2,9) ,DJACM(2,2)
      IF(NLAPS.GT.1) GO TO 10
C
C*** FOR SMALL DISPLACEMENT
C
      DJACM(1,1)=1.0
      DJACM(2,2)=1.0
      DJACM(1,2)=0.0
      DJACM(2,1)=0.0
      RETURN
C
C*** FOR LARGE DISPLACEMENT
C
   10 CONTINUE
      DO 20 IDIME=1,NDIME
      DO 20 JDIME=1,NDIME
      DJACM(IDIME,JDIME)=0.0
      DO 20 INODE=1,NNODE
      DJACM(IDIME,JDIME)=DJACM(IDIME,JDIME)
     .+DLCOD(IDIME,INODE)*CARTD(JDIME,INODE)
   20 CONTINUE
      RETURN
      END
```

10.6.14 Subroutine LINGNL

This routine calculates the total elastic strain and corresponding elastic stresses at a particular integration point. In this calculation the strains are evaluated using the deformation Jacobian matrix if geometric nonlinear behaviour is to be taken into account.

```
      SUBROUTINE LINGNL    (CARTD ,DJACM ,DMATX ,ELDIS ,GPCOD ,KGASP ,
     .                      KGAUS ,NDOFN ,NLAPS ,NNODE ,NSTRE ,NTYPE ,
     .                      POISS ,SHAPE ,STRAN ,STRES ,STRIN )
C*********************************************************************
C
C*** ELASTIC STRAIN AND STRESSES
C
C*********************************************************************
      DIMENSION CARTD(2,9) ,STRAN(4) ,DMATX(4,4) ,STRIN(4,1) ,
     .          ELDIS(2,9) ,STRES(4) ,DJACM(2,2) ,AGASH(2,2) ,
     .          GPCOD(2,9) ,SHAPE(9)
C
C*** CALCULATE STRAINS FROM DEFORMATION JACOBIAN
C
      IF(NLAPS.LT.2) GO TO 15
      STRAN(1)=0.5*(DJACM(1,1)*DJACM(1,1)+DJACM(2,1)*DJACM(2,1)-1.)
      STRAN(2)=0.5*(DJACM(1,2)*DJACM(1,2)+DJACM(2,2)*DJACM(2,2)-1.)
      STRAN(3)=DJACM(1,1)*DJACM(1,2)+DJACM(2,1)*DJACM(2,2)
C
C *** FOR SMALL DISPLACEMENTS
C
      GO TO 25
   15 CONTINUE
      DO 10 IDOFN=1,NDOFN
      DO 10 JDOFN=1,NDOFN
      BGASH=0.0
      DO 20 INODE=1,NNODE
   20 BGASH=BGASH+CARTD(JDOFN,INODE)*ELDIS(IDOFN,INODE)
   10 AGASH(IDOFN,JDOFN)=BGASH
      STRAN(1)=AGASH(1,1)
      STRAN(2)=AGASH(2,2)
      STRAN(3)=AGASH(1,2)+AGASH(2,1)
   25 CONTINUE
      IF(NTYPE.LT.3) GO TO 90
      STRAN(4)=0.0
      DO 70 INODE=1,NNODE
   70 STRAN(4)=STRAN(4)+ELDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP)
      EXTRA=0.0
      DO 80 INODE=1,NNODE
   80 EXTRA=EXTRA+ELDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP)
      STRAN(4)=STRAN(4)+0.5*EXTRA*EXTRA
   90 DO 50 ISTRE=1,4
      STRAN(ISTRE)=STRAN(ISTRE)-STRIN(ISTRE,KGAUS)
   50 CONTINUE
C
C*** AND THE CORRESPONDING STRESSES
C
      DO 30 ISTRE=1,NSTRE
      STRES(ISTRE)=0.0
      DO 30 JSTRE=1,NSTRE
   30 STRES(ISTRE)=STRES(ISTRE)+DMATX(ISTRE,JSTRE)*STRAN(JSTRE)
      IF(NTYPE.EQ.1) STRES(4)=0.0
      IF(NTYPE.EQ.2) STRES(4)=POISS*(STRES(1)+STRES(2))
      RETURN
      END
```

10.6.15 Subroutine LOADPL

This routine reads load data and evaluates the consistent nodal forces associated with thermal loading. A similar routine was described in Section 6.4.5. The additions which are included here have been discussed in detail in the authors' earlier text *Finite Element Programming*.[7]

```
      SUBROUTINE LOADPL    (COORD ,FORCE ,LNODS ,MATNO ,NDIME ,NDOFN ,
     .                      NELEM ,NGAUS ,NMATS ,NNODE ,NPOIN ,NSTRE ,
     .                      NTYPE ,POSGP ,PROPS ,RLOAD ,STRIN ,TEMPE ,
     .                      WEIGP )
C*********************************************************************
C
C*** STANDARD LOAD ROUTINE
C
C*********************************************************************
      DIMENSION COORD(NPOIN,1) ,GPCOD(2,9) ,POSGP(1) ,STRAN(4) ,
     .          LNODS(NELEM,1) ,CARTD(2,9) ,WEIGP(1) ,STRES(4) ,
     .          PROPS(NMATS,1) ,DERIV(2,9) ,TEMPE(1) ,NOPRS(3) ,
     .          RLOAD(NELEM,1) ,ELCOD(2,9) ,MATNO(1) ,DGASH(2) ,
     .          STRIN(   4, 1) ,PRESS(3,2) ,SHAPE(9) ,PGASH(2) ,
     .          DMATX(   4, 4) ,TITLE( 10) ,POINT(2) ,FORCE(1)
      TWOPI=6.283185307179586
      NEVAB=NNODE*NDOFN
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
   10 RLOAD(IELEM,IEVAB)=0.0
      READ(5,901) TITLE
  901 FORMAT (10A4)
      WRITE(6,903) TITLE
  903 FORMAT(/5X,17HLOAD CASE TITLE -,10A4)
C
C*** READ DATA CONTROLLING LOADING TYPES TO BE INPUTTED
C
      READ (5,919) IPLOD,IGRAV,IEDGE,ITEMP
      WRITE(6,990)
  990 FORMAT(/5X,21HLOAD INPUT PARAMETERS)
      WRITE(6,991) IPLOD,IGRAV,IEDGE,ITEMP
  991 FORMAT(/5X,12HPOINT LOADS ,I5/5X,12HGRAVITY     ,I5/
     .        5X,12HEDGE LOAD   ,I5/5X,12HTEMPERATURE ,I5)
  919 FORMAT(16I5)
C
C*** READ NODAL POINT LOADS
C
      IF(IPLOD.EQ.0) GO TO 500
      WRITE(6,998)
  998 FORMAT(/5X,5H NODE,10H        PX,10H        PY/)
   20 READ (5,931) LODPT,(POINT(IDOFN),IDOFN=1,NDOFN)
      WRITE(6,933) LODPT,(POINT(IDOFN),IDOFN=1,NDOFN)
  933 FORMAT(5X,I5,2G10.3)
  931 FORMAT(I5,2F10.3)
C
C*** ASSOCIATE THE NODAL POINT LOADS WITH AN ELEMENT
C
      DO 30 IELEM=1,NELEM
      DO 30 INODE=1,NNODE
      NLOCA=IABS(LNODS(IELEM,INODE))
   30 IF(LODPT.EQ.NLOCA) GO TO 40
   40 DO 50 IDOFN=1,NDOFN
      NGASH=(INODE-1)*NDOFN+IDOFN
   50 RLOAD(IELEM,NGASH)=POINT(IDOFN)
      IF(LODPT.LT.NPOIN) GO TO 20
  500 CONTINUE
      IF(IGRAV.EQ.0) GO TO 600
C
C*** READ GRAVITY ANGLE AND GRAVITATIONAL CONSTANT
C
      READ(5,906) THETA,GRAVY
  906 FORMAT(2F10.3)
      WRITE(6,911) THETA,GRAVY
  911 FORMAT(1H0,16H GRAVITY ANGLE =,F10.3,19H GRAVITY CONSTANT =,F10.3)
```

```
      THETA=THETA/57.295779514
C
      DO 90 IELEM=1,NELEM
C
C*** SET UP PRELIMINARY CONSTANTS
C
      LPROP=MATNO(IELEM)
      THICK=PROPS(LPROP,3)
      DENSE=PROPS(LPROP,4)
      IF(DENSE.EQ.0.0) GO TO 90
      GXCOM=DENSE*GRAVY*SIN(THETA)
      GYCOM=-DENSE*GRAVY*COS(THETA)
C
C*** COMPUTE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 60 INODE=1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      DO 60 IDIME=1,NDIME
   60 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      KGASP=0
      DO 80 IGAUS=1,NGAUS
      DO 80 JGAUS=1,NGAUS
      KGASP=KGASP+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
C
C*** COMPUTE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS AND ELEMENTAL
C    VOLUME
C
      CALL SFR2   (DERIV,NNODE,SHAPE,EXISP,ETASP)
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .             KGASP,NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.1) DVOLU=DVOLU*THICK
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
C
C*** CALCULATE LOADS AND ASSOCIATE WITH ELEMENT NODAL POINTS
C
      DO 70 INODE=1,NNODE
      NGASH=(INODE-1)*NDOFN+1
      MGASH=(INODE-1)*NDOFN+2
      RLOAD(IELEM,NGASH)=RLOAD(IELEM,NGASH)+GXCOM*SHAPE(INODE)*DVOLU
   70 RLOAD(IELEM,MGASH)=RLOAD(IELEM,MGASH)+GYCOM*SHAPE(INODE)*DVOLU
   80 CONTINUE
   90 CONTINUE
  600 CONTINUE
      IF(IEDGE.EQ.0) GO TO 700
C
C*** DISTRIBUTED EDGE LOADS SECTION
C
      READ(5,932) NEDGE
  932 FORMAT(I5)
      WRITE(6,912) NEDGE
  912 FORMAT(1H0,5X,21HNO. OF LOADED EDGES =,I5)
      WRITE(6,915)
  915 FORMAT(1H0,5X,38HLIST OF LOADED EDGES AND APPLIED LOADS)
      NODEG=3
      NCODE=NNODE
      IF(NNODE.EQ.4) NODEG=2
      IF(NNODE.EQ.9) NCODE=8
C
```

```
C*** LOOP OVER EACH LOADED EDGE
C
      DO 160 IEDGE=1,NEDGE
C
C*** READ DATA LOCATING THE LOADED EDGE AND APPLIED LOAD
C
      READ (5,902) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  902 FORMAT(4I5)
      WRITE(6,913) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  913 FORMAT(I10,5X,3I5)
      READ (5,914) ((PRESS(IODEG,IDOFN),IODEG=1,NODEG),IDOFN=1,NDOFN)
      WRITE(6,914) ((PRESS(IODEG,IDOFN),IODEG=1,NODEG),IDOFN=1,NDOFN)
  914 FORMAT(6F10.3)
      ETASP=-1.0
C
C*** CALCULATE THE COORDINATES OF THE NODES OF THE ELEMENT EDGE
C
      DO 100 IODEG=1,NODEG
      LNODE=NOPRS(IODEG)
      DO 100 IDIME=1,NDIME
  100 ELCOD(IDIME,IODEG)=COORD(LNODE,IDIME)
C
C*** ENTER LOOP FOR LINEAR NUMERICAL INTEGRATION
      DO 150 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS
C
      CALL SFR2   (DERIV,NNODE,SHAPE,EXISP,ETASP)
C
C*** CALCULATE COMPONENTS OF THE EQUIVALENT NODAL LOADS
C
      DO 110 IDOFN=1,NDOFN
      PGASH(IDOFN)=0.0
      DGASH(IDOFN)=0.0
      DO 110 IODEG=1,NODEG
      PGASH(IDOFN)=PGASH(IDOFN)+PRESS(IODEG,IDOFN)*SHAPE(   IODEG)
  110 DGASH(IDOFN)=DGASH(IDOFN)+ELCOD(IDOFN,IODEG)*DERIV(1,IODEG)
      DVOLU=WEIGP(IGAUS)
      PXCOM=DGASH(1)*PGASH(2)-DGASH(2)*PGASH(1)
      PYCOM=DGASH(1)*PGASH(1)+DGASH(2)*PGASH(2)
      IF(NTYPE.NE.3) GO TO 115
      RADUS=0.0
      DO 125 IODEG=1,NODEG
  125 RADUS=RADUS+SHAPE(IODEG)*ELCOD(1,IODEG)
      DVOLU=DVOLU*TWOPI*RADUS
  115 CONTINUE
C
C*** ASSOCIATE THE EQUIVALENT NODAL EDGE LOADS WITH AN ELEMENT
C
      DO 120 INODE=1,NNODE
      NLOCA=IABS(LNODS(NEASS,INODE))
  120 IF(NLOCA.EQ.NOPRS(1)) GO TO 130
  130 JNODE=INODE+NODEG-1
      KOUNT=0
      DO 140 KNODE=INODE,JNODE
      KOUNT=KOUNT+1
      NGASH=(KNODE-1)*NDOFN+1
      MGASH=(KNODE-1)*NDOFN+2
      IF(KNODE.GT.NCODE) NGASH=1
      IF(KNODE.GT.NCODE) MGASH=2
      RLOAD(NEASS,NGASH)=RLOAD(NEASS,NGASH)+SHAPE(KOUNT)*PXCOM*DVOLU
  140 RLOAD(NEASS,MGASH)=RLOAD(NEASS,MGASH)+SHAPE(KOUNT)*PYCOM*DVOLU
  150 CONTINUE
  160 CONTINUE
```

```
  700 CONTINUE
      IF(ITEMP.EQ.0) GO TO 800
C
C*** INITIALIZE AND INPUT THE NODAL TEMPERATURES
C
      DO 170 IPOIN=1,NPOIN
  170 TEMPE(IPOIN)=0.0
      WRITE(6,917)
  917 FORMAT(1H0,5X,29HPRESCRIBED NODAL TEMPERATURES)
  180 READ (5,916) NODPT,TEMPE(NODPT)
      WRITE(6,916) NODPT,TEMPE(NODPT)
  916 FORMAT(I5,F10.3)
      IF(NODPT.LT.NPOIN) GO TO 180
      KGAST=0
C
C*** LOOP OVER EACH ELEMENT
C
      DO 280 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      DO 200 INODE=1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
C
C*** IDENTIFY THE COORDINATES AND TEMPERATURE OF EACH ELEMENT NODE POINT
C
      DO 190 IDIME=1,NDIME
  190 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
  200 ELCOD(2,INODE)=TEMPE(LNODE)
C
C*** SET UP MATERIAL PROPERTIES
C
      CALL MODPS  (DMATX,LPROP,NMATS,NSTRE,NTYPE,PROPS)
      YOUNG=PROPS(LPROP,1)
      POISS=PROPS(LPROP,2)
      THICK=PROPS(LPROP,3)
      ALPHA=PROPS(LPROP,5)
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      KGASP=0
      DO 270 IGAUS=1,NGAUS
      DO 270 JGAUS=1,NGAUS
      KGAST=KGAST+1
      KGASP=KGASP+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS AND TEMPERATURE AT THE SAMPLING POINTS
C    ,ELEMENTAL VOLUME AND CARTESIAN DERIVATIVES
C
      CALL SFR2   (DERIV,NNODE,SHAPE,EXISP,ETASP)
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .             KGASP,NNODE,SHAPE)
      THERM=0.0
      DO 210 INODE=1,NNODE
  210 THERM=THERM+ELCOD(2,INODE)*SHAPE(INODE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.1) DVOLU=DVOLU*THICK
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
C
C*** EVALUATE THE INITIAL THERMAL STRAINS
C
      EIGEN=THERM*ALPHA
      IF(NTYPE.EQ.2) GO TO 220
      STRAN(1)=-EIGEN
      STRAN(2)=-EIGEN
```

```
      STRAN(3)=0.0
      GO TO 230
  220 STRAN(1)=-(1.0+POISS)*EIGEN
      STRAN(2)=-(1.0+POISS)*EIGEN
      STRAN(3)=0.0
C
C*** AND THE CORRESPONDING INITIAL STRESSES
C
  230 DO 250 ISTRE=1,NSTRE
      STRES(ISTRE)=0.0
      DO 240 JSTRE=1,NSTRE
  240 STRES(ISTRE)=STRES(ISTRE)+DMATX(ISTRE,JSTRE)*STRAN(JSTRE)
  250 STRIN(ISTRE,KGAST)=STRES(ISTRE)
      IF(NTYPE.EQ.2) STRIN(4,KGAST)=-YOUNG*EIGEN
      IF(NTYPE.EQ.1) STRIN(4,KGAST)=0.0
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE
C    ELEMENT NODES
C
      EXTRA=0.0
      DO 260 INODE=1,NNODE
      IF(NTYPE.EQ.3) EXTRA=DVOLU*SHAPE(INODE)*STRES(4)/GPCOD(1,KGASP)
      NGASH=(INODE-1)*NDOFN+1
      MGASH=(INODE-1)*NDOFN+2
      RLOAD(IELEM,NGASH)=RLOAD(IELEM,NGASH)+EXTRA
     . -(CARTD(1,INODE)*STRES(1)+CARTD(2,INODE)*STRES(3))*DVOLU
  260 RLOAD(IELEM,MGASH)=RLOAD(IELEM,MGASH)
     . -(CARTD(1,INODE)*STRES(3)+CARTD(2,INODE)*STRES(2))*DVOLU
  270 CONTINUE
  280 CONTINUE
  800 CONTINUE
C     WRITE(6,907)
C 907 FORMAT(1H0,5X,36H TOTAL NODAL FORCES FOR EACH ELEMENT)
C     DO 290 IELEM=1,NELEM
C 290 WRITE(6,905) IELEM,(RLOAD(IELEM,IEVAB),IEVAB=1,NEVAB)
C 905 FORMAT(1X,I4,5X,8E12.4/(10X,8E12.4))
      DO 5 IELEM=1,NELEM
      KEVAB=0
      DO 5 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      NPOSN=(LNODE-1)*NDOFN
      DO 5 IDOFN=1,NDOFN
      KEVAB=KEVAB+1
      NPOSN=NPOSN+1
      FORCE(NPOSN)=FORCE(NPOSN)+RLOAD(IELEM,KEVAB)
    5 CONTINUE
      RETURN
      END
```

10.6.16 Subroutine LUMASS

This subroutine evaluates the lumped mass vector and consistent mass matrix for the finite element mesh. If INTGR(I) = 1, it generates the consistent mass matrix and if INTGR(I) = 2, it generates a special lumped mass vector. In the special mass lumping scheme which is employed, the diagonal terms of the consistent mass matrix are scaled to preserve the total mass. The element consistent mass matrices are written on tape 3. The consistent mass matrix is not used in DYNPAK.

This subroutine also reads concentrated masses and assembles them into the global diagonal mass vector.

```
      SUBROUTINE LUMASS (COORD ,INTGR ,LNODS ,MATNO ,NCONM ,NDIME ,
     .                   NDOFN ,NELEM ,NGAUM ,NMATS ,NNODE ,NPOIN ,
     .                   NTYPE ,PROPS ,YMASS )
C*********************************************************************
C
C *** CALCULATES LUMPED MASS FOR 4 , 8 AND 9 NODED ELEMENT
C
C*********************************************************************
      DIMENSION COORD(NPOIN,1) ,ELCOD(2,9) ,DIAGM(9) ,POSGP(4) ,
     .          LNODS(NELEM,1) ,CARTD(2,9) ,SHAPE(9) ,WEIGP(4) ,
     .          PROPS(NMATS,1) ,GPCOD(2,9) ,MATNO(1) ,YMASS(1) ,
     .          EMASS(    171) ,DERIV(2,9) ,INTGR(1)
C
      REWIND 3
      TWOPI=6.283185307179586
      NEVAB=NNODE*NDOFN
      NTOTV=NPOIN*NDOFN
      DO 500 ITOTV =1,NTOTV
  500 YMASS(ITOTV)=0.0
      CALL GAUSSQ (NGAUM ,POSGP ,WEIGP )
      DO 100 IELEM=1,NELEM
      DO 5 IEVAB=1,171
    5 EMASS(IEVAB)=0.0
      IMASS=INTGR(IELEM)
      KGASP=0
      TAREA=0.0
      LPROP=MATNO(IELEM)
      THICK=PROPS(LPROP,3)
      RHOEL=PROPS(LPROP,4)
      DO 10 INODE=1,NNODE
      DIAGM(INODE)=0.0
      LNODE=LNODS(IELEM,INODE)
      DO 10 IDIME=1,NDIME
      ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
   10 CONTINUE
      DO 70 IGAUS=1,NGAUM
      EXISP=POSGP(IGAUS)
      DO 70 JGAUS=1,NGAUM
      KGASP=KGASP+1
      ETASP=POSGP(JGAUS)
      CALL       SFR2   (DERIV,NNODE,SHAPE,EXISP,ETASP)
      CALL       JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,
     .                   KGASP,NNODE,SHAPE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.1) DVOLU=DVOLU*THICK
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(IMASS.EQ.1) GO TO 210
      DO 20 INODE=1,NNODE
      SHAPI=SHAPE(INODE)
   20 DIAGM(INODE)=DIAGM(INODE)+SHAPI*SHAPI*DVOLU
      TAREA=TAREA+DVOLU
  210 IF(IMASS.EQ.2) GO TO 70
      DVOLU=DVOLU*RHOEL
      IEVAB=1
      KOUNT=NEVAB
      DO 30 INODE=1,NNODE
      SHAPI=SHAPE(INODE)
      DO 60 JNODE=INODE,NNODE
      DMASS=DVOLU*SHAPI*SHAPE(JNODE)
      EMASS(IEVAB)=EMASS(IEVAB)+DMASS
      JEVAB=IEVAB+KOUNT
      EMASS(JEVAB)=EMASS(JEVAB)+DMASS
   60 IEVAB=IEVAB+2
      KOUNT=KOUNT-2
```

```
      IEVAB=JEVAB+1                                              MASS  65
   30 CONTINUE                                                   MASS  66
   70 CONTINUE                                                   MASS  67
C                                                                MASS  68
C***  WRITES CONSISTENT MASS MATRIX ON TAPE 3                    MASS  69
C                                                                MASS  70
      IF(IMASS.EQ.2) GO TO 200                                   MASS  71
      WRITE(3) EMASS                                             MASS  72
C     WRITE(6,90) (EMASS(I),I=1,171)                             MASS  73
  200 IF(IMASS.EQ.1) GO TO 100                                   MASS  74
C                                                                MASS  75
C *** GENERATES LUMPED MASS MATRIX PROPORTIONAL TO DIAGONAL      MASS  76
C                                                                MASS  77
      SUMAS=0.                                                   MASS  78
      DO 40 INODE=1,NNODE                                        MASS  79
   40 SUMAS=SUMAS+DIAGM(INODE)                                   MASS  80
      TAREA=TAREA*RHOEL                                          MASS  81
      SUMAS=TAREA/SUMAS                                          MASS  82
      DO 50 INODE=1,NNODE                                        MASS  83
      LNODE=LNODS(IELEM,INODE)                                   MASS  84
      IPOSN=(LNODE-1)*NDOFN                                      MASS  85
      DO 50 IDOFN=1,NDOFN                                        MASS  86
      IPOSN=IPOSN+1                                              MASS  87
      YMASS(IPOSN)=YMASS(IPOSN)+DIAGM(INODE)*SUMAS               MASS  88
   50 CONTINUE                                                   MASS  89
   90 FORMAT(2X,9E12.3)                                          MASS  90
  100 CONTINUE                                                   MASS  91
C                                                                MASS  92
C     CONCENTRATED MASSES                                        MASS  93
C                                                                MASS  94
      IF(NCONM.EQ.0) RETURN                                      MASS  95
      WRITE(6,900)                                               MASS  96
      DO 520 ICONM=1,NCONM                                       MASS  97
      READ(5,910) IPOIN,XCMAS,YCMAS                              MASS  98
  900 FORMAT(5X,19HCONCENTRATED MASSES)                          MASS  99
      WRITE(6,910) IPOIN,XCMAS,YCMAS                             MASS 100
      NPOSN=(IPOIN-1)*NDOFN+1                                    MASS 101
      YMASS(NPOSN)=YMASS(NPOSN)+XCMAS                            MASS 102
      NPOSN=NPOSN+1                                              MASS 103
      YMASS(NPOSN)=YMASS(NPOSN)+YCMAS                            MASS 104
  520 CONTINUE                                                   MASS 105
C     WRITE(6,90) (YMASS(I),I=1,NTOTV)                           MASS 106
  910 FORMAT(I5,2F10.3)                                          MASS 107
      RETURN                                                     MASS 108
      END                                                        MASS 109
```

MASS 24	Sets indicator for mass matrix evaluation. INTGR(I) = 1 for the consistent mass matrix and INTGR(I) = 2 for the special lumped mass vector.
MASS 35–52	Evaluate the diagonal element of the consistent mass matrix DIAGM.
MASS 53–63	Evaluates the element consistent mass matrix.
MASS 72	Writes element consistent mass matrix on tape 3.
MASS 78–80	Evaluates ELMAS, the sum of the diagonal elements.
MASS 81	Determines the total element mass from the element volume TAREA and mass density RHOEL.

MASS 83–89 Scales the diagonal terms using the factor TAREA/ELMAS to preserve element mass and assembles the result into diagonal mass vector YMASS.

MASS 95–107 Reads the concentrated masses and assembles them into YMASS.

10.6.17 Subroutine MODPS

This subroutine evaluates the elasticity matrix and has been described earlier in Chapter 6. The only changes involved are given below.

```
      SUBROUTINE MODPS     (DMATX ,LPROP ,NMATS ,NSTRE ,NTYPE ,PROPS )
C***********************************************************************
C
C **   ELASTICITY D MATRIX
C
C***********************************************************************
      DIMENSION DMATX(4,4),PROPS(NMATS,1)
```

10.6.18 Subroutine NODXYR

It calculates (r, z) coordinates from (R, Θ) coordinates for axisymmetric problems. If coordinates of midside nodes are not read, it evaluates them by linear interpolation. An almost identical subroutine was described in Chapter 6.

```
      SUBROUTINE NODXYR (COORD,LNODS,NELEM,NNODE,NPOIN,NRADS,NTYPE)
C***********************************************************************
C
C*** INTERPOLATION OF MIDSIDE AND CENTER NODES
C
C***********************************************************************
      DIMENSION  COORD(NPOIN,1),LNODS(NELEM,1)
C
      IF(NTYPE.NE.3.OR.NRADS.EQ.0) GO TO 40
C
C*** CHANGE POLAR COORDINATES TO CARTISIAN
      DO 50 IPOIN=1,NPOIN
      RADDI=COORD(IPOIN,1)
      THETA=COORD(IPOIN,2)
      THETA=0.017453292*THETA
      COORD(IPOIN,1)=RADDI*SIN(THETA)
   50 COORD(IPOIN,2)=RADDI*COS(THETA)
C
   40 IF(NNODE.EQ.4) RETURN
C
      LNODE = NNODE - 1
      DO 30 IELEM=1,NELEM
C*** LOOP OVER EACH ELEMENT EDGE
      DO 20 INODE=1,NNODE,2
      IF(INODE.EQ.9) GO TO 20
C*** COMPUTE THE NODE NUMBER OF THE FIRST NODE
      NODST=LNODS(IELEM,INODE)
      IGASH=INODE+2
      IF(IGASH.GT.LNODE) IGASH=1
C*** COMPUTE THE NODE NUMBER OF THE LAST NODE
      NODFN=LNODS(IELEM,IGASH)
      MIDPT=INODE+1
```

```
C*** COMPUTE THE NODE NUMBER OF THE INTERMEDIATE NODE
      NODMD=LNODS(IELEM,MIDPT)
      TOTAL=ABS(COORD(NODMD,1))+ABS(COORD(NODMD,2))
C*** IF THE COORDINATES OF THE INTERMEDIATE NODE ARE BOTH ZERO
C    INTERPOLATE BY A STRAIGHT LINE
      IF(TOTAL.GT.0.0) GO TO 20
      KOUNT=1
   10 COORD(NODMD,KOUNT)=(COORD(NODST,KOUNT)+COORD(NODFN,KOUNT))/2.0
      KOUNT=KOUNT+1
      IF(KOUNT.EQ.2) GO TO 10
   20 CONTINUE
   30 CONTINUE
      RETURN
      END
```

10.6.19 Subroutine OUTDYN

This routine writes out most of the output on the line printer and on various tapes for plotting purposes. It outputs the displacements and stresses every NOUTP steps. It also writes the displacement and stress histories of specified nodal and integration points at every NOUTP steps. The complete state of displacements is also written on tape 13 for a deformation plot. The complete state of the stresses is written on tape 4. The principal stresses and their directions are also calculated and output.

```
      SUBROUTINE OUTDYN    (DISPL ,DTIME ,ISTEP ,NDOFN ,NELEM ,NGAUS ,
     .                      NGRQS ,NOUTD ,NOUTP ,NPOIN ,NPRQD ,NREQD ,
     .                      NREQS ,NTYPE ,STRSG ,TDISP ,VIVEL )
C*********************************************************************
C
C** OUTPUT ROUTINE
C
C*********************************************************************
      DIMENSION STRSG(4,1) ,DISPL(1) ,NPRQD(1) ,STRSP(3) ,
     .          VIVEL(5,1) ,TDISP(1) ,NGRQS(1)
      NSTR1=4
      KSTEP=ISTEP
      MGAUS=NELEM*NGAUS*NGAUS
      IF(ISTEP.EQ.1) WRITE(10,925)
      TTIME=TTIME+DTIME
C
C ***  WRITES DISPLACEMENT HISTORY AT REQUESTED NODAL POINTS ON TAPE 10
C ***  AND STRESS HISTORY AT REQUESTED GAUSS POINTS AT EVERY NOUTD STEPS
C
      KOUNT=0
      KOUTD=(ISTEP/NOUTD)*NOUTD
      IF(KOUTD.NE.ISTEP) GO TO 510
      DO 500 IPOIN=1,NPOIN
      DO 500 IREQD=1,NREQD
      IF(IPOIN.NE.NPRQD(IREQD)) GO TO 500
      NPOSN=(IPOIN-1)*NDOFN+1
      NPOSM=NPOSN+1
      KOUNT=KOUNT+1
      DISPL(KOUNT)=TDISP(NPOSN)
      KOUNT=KOUNT+1
      DISPL(KOUNT)=TDISP(NPOSM)
  500 CONTINUE
      WRITE(10,960) (DISPL(IKOUN),IKOUN=1,KOUNT),TTIME
```

```
      DO 520 IGAUS=1,MGAUS
      DO 520 IREQS=1,NREQS
      IF(IGAUS.NE.NGRQS(IREQS)) GO TO 520
      WRITE(11,950) (STRSG(ISTR1,IGAUS),ISTR1=1,NSTR1)
  520 CONTINUE
  510 KOUTD=(KSTEP/NOUTP)*NOUTP
      IF(KOUTD.NE.KSTEP) RETURN
      XTIME=FLOAT(KSTEP)*DTIME
      WRITE(6,604) KSTEP,XTIME
  604 FORMAT(//5X,28H DISPLACEMENTS AT TIME STEP ,I10,5X,5HTIME ,E20.11)
C
C *** REARRANGE DISPLACEMENT VECTOR
C
      NODEI=0
      DO 550 IPOIN=1,NPOIN
      DO 550 IDOFN=1,NDOFN
      NODEI=NODEI+1
      DISPL(NODEI)=TDISP(NODEI)
  550 CONTINUE
C
C*** OUTPUT DISPLACEMENTS
C
  925 FORMAT(5X,' DISPLACEMENTS ')
      WRITE(6,990)
  990 FORMAT(/3(1X,'NNODE',3X,'X-DISP',6X,'Y-DISP',3X)/)
      DO 560 IPOIN=1,NPOIN,3
      NGASI=NDOFN*IPOIN-1
      NGASJ=NGASI+NDOFN
      NGASK=NGASJ+NDOFN
      MGASI=NGASI+1
      MGASJ=NGASJ+1
      MGASK=NGASK+1
      JPOIN=IPOIN+1
      KPOIN=JPOIN+1
C
C *** WRITES DISPLACEMENTS  ON TAPE 13 FOR DEFORMATION PLOT
C
      WRITE(13,910) IPOIN ,(DISPL(IGASI),IGASI=NGASI,MGASI)
      IF(JPOIN.GT.NPOIN) GO TO 200
      WRITE(13,910) JPOIN ,(DISPL(IGASJ),IGASJ=NGASJ,MGASJ)
      IF(KPOIN.GT.NPOIN) GO TO 200
      WRITE(13,910) KPOIN ,(DISPL(IGASK),IGASK=NGASK,MGASK)
  200 CONTINUE
C
C *** WRITES DISPLACEMENTS ON OUTPUT FILE
C
  560 WRITE(6,920) IPOIN,DISPL(NGASI),DISPL(MGASI),
     .             JPOIN,DISPL(NGASJ),DISPL(MGASJ),
     .             KPOIN,DISPL(NGASK),DISPL(MGASK)
C
C *** WRITES STRESSES ON OUTPUT FILE
C
      WRITE(6,900)
      IF(NTYPE.NE.3) WRITE(6,970)
  970 FORMAT(1H0,1X,4HG.P.,6X,9HXX-STRESS,5X,9HYY-STRESS,5X,9HXY-STRESS,
     .5X,9HZZ-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,5HANGLE,3X,6H  P.S.)
      IF(NTYPE.EQ.3) WRITE(6,975)
  975 FORMAT(1H0,1X,4HG.P.,6X,9HRR-STRESS,5X,9HZZ-STRESS,5X,9HRZ-STRESS,
     .5X,9HTT-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,5HANGLE,3X,6H  P.S.)
      KGAUS=0
      DO 570 IELEM=1,NELEM
      KELGS=0
      WRITE(6,930) IELEM
  930 FORMAT(1H0,5X,13HELEMENT NO. =,I5)
```

```
      DO 570 IGAUS=1,NGAUS
      DO 570 JGAUS=1,NGAUS
      KGAUS=KGAUS+1
      KELGS=KELGS+1
      XGASH=(STRSG(1,KGAUS)+STRSG(2,KGAUS))*0.5
      XGISH=(STRSG(1,KGAUS)-STRSG(2,KGAUS))*0.5
      XGESH=STRSG(3,KGAUS)
      XGOSH=SQRT(XGISH*XGISH+XGESH*XGESH)
      STRSP(1)=XGASH+XGOSH
      STRSP(2)=XGASH-XGOSH
      IF(XGISH.EQ.0.0) XGISH=0.1E-20
      STRSP(3)=ATAN(XGESH/XGISH)*28.647889757
C
C *** WRITES COMPLETE STRESS STATE ON TAPE 4
C
      WRITE(4,950) (STRSG(ISTR1,KGAUS),ISTR1=1,NSTR1),
     .(STRSP(ISTRE),ISTRE=1,3)
  570 WRITE(6,940) KELGS,(STRSG(ISTR1,KGAUS),ISTR1=1,NSTR1),
     .(STRSP(ISTRE),ISTRE=1,3),VIVEL(5,KGAUS)
  980 FORMAT(1X,60I2)
  960 FORMAT(1X,10E11.4)
  950 FORMAT (7E10.4)
  940 FORMAT(I5,2X,6E14.6,F8.3,E14.6)
  900 FORMAT(/,10X,8HSTRESSES,/)
  920 FORMAT(3(1X,I5,2E12.5))
  910 FORMAT(I5,2E15.6)
      RETURN
      END
```

10.6.20 Subroutine PREVOS

This routine reads and write the initial forces and stresses.

```
      SUBROUTINE PREVOS (FORCE ,NDOFN ,NELEM ,NGAUS ,NPOIN ,NPREV ,
     .                   STRIN )
C*********************************************************************
C
C***   GRAVITY LOADS AND STRESSES
C
C*********************************************************************
      DIMENSION FORCE(1) ,STRIN(4,1)
C
      IF(NPREV.EQ.0) RETURN
C
      NSTR1=4
      NGAU2=NGAUS*NGAUS
C
C*** READ  GRAVITY LOADS
C
      WRITE(6,920)
  920 FORMAT(//4X,6H NODE ,17H GRAVITY X-LOAD: ,17H GRAVITY Y-LOAD: /)
  200 READ (5,900) NGASH,XGASH,YGASH
  900 FORMAT(I5,4F10.3)
  910 FORMAT(I10,4E18.5)
      NPOSN=(NGASH-1)*NDOFN+1
      FORCE(NPOSN)=XGASH
      NPOSN=NPOSN+1
      FORCE(NPOSN)=YGASH
      WRITE(6,910) NGASH,XGASH,YGASH
      IF (NGASH.NE.NPOIN) GO TO 200
C
C*** READ  GRAVITY STRESS
C
```

```
      WRITE(6,930)
  930 FORMAT(//2X,9HGAUSS PT.,17H GRAVITY X-STRESS,17H GRAVITY Y-STRESS,
     .18H GRAVITY XY-STRESS,17H GRAVITY Z-STRESS/)
      DO 500 IELEM=1,NELEM
      DO 500 IGAUS=1,NGAU2
      READ(5,900) KGAUS,(STRIN(ISTRI,KGAUS),ISTRI=1,NSTR1)
  500 WRITE(6,910)KGAUS,(STRIN(ISTRI,KGAUS),ISTRI=1,NSTR1)
      RETURN
      END
```

10.6.21 Subroutine RESVPL

This routine evaluates the internal resisting force vector

$$p_n = \int_{\Omega} [B]_n{}^T \sigma_n \, d\Omega.$$

It is very similar to the routine described in Section 8.8.

```
      SUBROUTINE RESVPL (COORD ,DTIME ,LNODS ,MATNO ,NCRIT ,NDIME ,
     .                   NDOFN ,NELEM ,NGAUS ,NLAPS ,NNODE ,NMATS ,
     .                   NPOIN ,NSTRE ,NTYPE ,POSGP ,PROPS ,RESID ,
     .                   RLOAD ,STRIN ,STRSG ,TDISP ,VISTN ,VIVEL ,
     .                   WEIGP )
C*******************************************************************
C
C***  EVALUATION OF INTEGRAL (B)**T*(SIGMA)
C
C*******************************************************************
      DIMENSION COORD(NPOIN,1),DERIV(2,9),DJACM(2,2),AVECT(4),MATNO(1),
     .          PROPS(NMATS,1),DLCOD(2,9),STRIN(4,1),DEVIA(4),TDISP(1),
     .          LNODS(NELEM,1),GPCOD(2,9),STRSG(4,1),STRAN(4),POSGP(1),
     .          RLOAD(NELEM,1),CARTD(2,9),VISTN(4,1),STRES(4),WEIGP(1),
     .          DMATX(    4,4),ELCOD(2,9),VIVEL(5,1),SHAPE(9),RESID(1),
     .          BMATX(   4,18),ELDIS(2,9),DESTN(  4)
      KGAUS=0
      NSTR1=4
      NEVAB=NNODE*NDOFN
      NTOTV=NPOIN*NDOFN
      TWOPI=6.283185307179586
      DO 530 IELEM=1,NELEM
      DO 540 IEVAB=1,NEVAB
  540 RLOAD(IELEM,IEVAB)=0.0
  530 CONTINUE
      DO 510 ITOTV=1,NTOTV
  510 RESID(ITOTV)=0.0
C
C***  LOOP OVER ALL THE ELEMENTS
C
      DO 20 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      THICK=PROPS(LPROP,3)
      POISS=PROPS(LPROP,2)
      FRICT=PROPS(LPROP,8)
C
C*** COMPUTE NEW COORDINATES AND  DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 30 INODE =1,NNODE
      LNODE=IABS(LNODS(IELEM,INODE))
      NPOSN=(LNODE-1)*NDOFN
```

```
      DO 30 IDOFN=1,NDOFN
      NPOSN=NPOSN+1
      ELCOD(IDOFN,INODE)=COORD(LNODE,IDOFN)
      DLCOD(IDOFN,INODE)=COORD(LNODE,IDOFN)+TDISP(NPOSN)
   30 ELDIS(IDOFN,INODE)=TDISP(NPOSN)
      CALL MODPS  (DMATX,LPROP,NMATS,NSTRE,NTYPE,PROPS)
      KGASP=0
      DO 40 IGAUS=1,NGAUS
      DO 40 JGAUS=1,NGAUS
      KGAUS=KGAUS+1
      KGASP=KGASP+1
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
C
      CALL SFR2   (DERIV ,NNODE ,SHAPE ,EXISP ,ETASP )
      CALL JACOB2 (CARTD ,DERIV ,DJACB ,ELCOD ,GPCOD ,
     .             IELEM ,KGASP ,NNODE ,SHAPE )
      CALL JACOBD (CARTD ,DLCOD ,DJACM ,NDIME ,NLAPS ,NNODE )
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.1) DVOLU=DVOLU*THICK
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      CALL BLARGE (BMATX ,CARTD ,DJACM ,DLCOD ,GPCOD ,
     .             KGASP ,NLAPS ,NNODE ,NTYPE ,SHAPE )
      CALL LINGNL (CARTD ,DJACM ,DMATX ,ELDIS ,GPCOD ,KGASP,
     .             KGAUS ,NDOFN ,NLAPS ,NNODE ,NSTRE ,NTYPE,
     .             POISS ,SHAPE ,STRAN ,STRES ,VISTN )
C
      DO 580 ISTR1=1,NSTR1
  580 STRES(ISTR1)=STRES(ISTR1)+STRIN(ISTR1,KGAUS)
      DO 570 ISTR1=1,NSTR1
  570 STRSG(ISTR1,KGAUS)=STRES(ISTR1)
C
      IF(NLAPS.EQ.2.OR.NLAPS.EQ.0) GO TO 200
C
      CALL INVAR  (DEVIA,LPROP,NCRIT,NMATS,PROPS,SINT3,STEFF,
     .             STRES,THETA,VARJ2,YIELD)
      CALL YIELDF (AVECT,DEVIA,FRICT,NCRIT,SINT3,STEFF,THETA,VARJ2)
      CALL FLOWVP (AVECT,KGAUS,LPROP,NCRIT,NMATS,PROPS,
     .             STEFF,VIVEL,YIELD)
C
C***  VISCOPLASTIC STRAIN INCREMENT AND A MEASURE FOR HARDENING
C
      DO 60 ISTR1=1,NSTR1
      DESTN(ISTR1)=VIVEL(ISTR1,KGAUS)*DTIME
   60 VISTN(ISTR1,KGAUS)=VISTN(ISTR1,KGAUS)+DESTN(ISTR1)
      DEBAR=SQRT((2.0*(DESTN(1)*DESTN(1)+DESTN(2)*DESTN(2)+
     . DESTN(4)*DESTN(4))+DESTN(3)*DESTN(3))/3.0)
      VIVEL(5,KGAUS)=DEBAR
C
C***  COMPUT INT(B**T*SIGMA) ON ELEMENT LEVEL
C
  200 CONTINUE
      KEVAB=0
      DO 502 INODE=1,NNODE
      DO 502 IDOFN=1,NDOFN
      KEVAB=KEVAB+1
      DO 501 ISTRE=1,NSTRE
  501 RLOAD(IELEM,KEVAB)=RLOAD(IELEM,KEVAB)+
     .BMATX(ISTRE,KEVAB)*STRSG(ISTRE,KGAUS)*DVOLU
  502 CONTINUE
   40 CONTINUE
   20 CONTINUE
C
C***  ASSEMBLY OF RESID VECTOR
```

```
RESD 107 C
RESD 108       DO 500 IELEM=1,NELEM
RESD 109       KEVAB=0
RESD 110       DO 500 INODE=1,NNODE
RESD 111       LNODE=LNODS(IELEM,INODE)
RESD 112       NPOSN=(LNODE-1)*NDOFN
RESD 113       DO 500 IDOFN=1,NDOFN
RESD 114       KEVAB=KEVAB+1
RESD 115       NPOSN=NPOSN+1
RESD 116       RESID(NPOSN)=RESID(NPOSN)+RLOAD(IELEM,KEVAB)
RESD 117   500 CONTINUE
RESD 118       RETURN
RESD 119       END
```

RESD 66–68	Call LINGNL to determine the state of stress at the current Gauss point.
RESD 77–78	Call INVAR to evaluate stress invariants at the current Gauss point.
RESD 79	Call YIELDF to select the yield function and calculate the $\boldsymbol{a}$ vector.
RESD 80–81	Call FLOWVP to define the rate of viscoplastic straining VIVEL if the stress point is outside the current yield surface.
RESD 86	Evaluate the increments of viscoplastic strains DESTN.
RESD 87	Evaluate the viscoplastic strains $(\epsilon_{vp})_{n+1}$ for the next time station $t_n+\Delta t$, VISTN.
RESD 88–90	Determine a measure of hardening for the current yield surface.
RESD 95–101	Evaluate $\boldsymbol{p}_n{}^{(e)}$ at the element level, RLOAD.
RESD 108–117	Assemble $\boldsymbol{p}_n$, RESID.

10.6.22 Subroutine YIELDF

This subroutine selects the yield function and calculates the vector $\boldsymbol{a}$ (AVECT) and is almost identical to the version described in Section 7.8.4.1.

```
      SUBROUTINE YIELDF (AVECT ,DEVIA ,FRICT ,NCRIT ,SINT3 ,STEFF ,
     .                   THETA ,VARJ2 )
C***********************************************************************
C
C *** SELECTS YIELD FUNCTION AND CALCULATES VECTOR 'AVECT'
C
C***********************************************************************
      DIMENSION AVECT(4) ,DEVIA(4) ,VECA1(4) ,VECA2(4) ,VECA3(4)
      IF(STEFF.EQ.0.0) RETURN
      NSTR1=4
      TANTH=TAN(THETA)
      SINTH=SIN(THETA)
      COSTH=COS(THETA)
      COST3=COS(3.0*THETA)
      ROOT3=1.73205080757
```

```
C*** CALCULATE VECTOR A1
      VECA1(1)=1.0
      VECA1(2)=1.0
      VECA1(3)=0.0
      VECA1(4)=1.0
C*** CALCULATE VECTOR A2
      DO 10 ISTR1=1,NSTR1
   10 VECA2(ISTR1)=DEVIA(ISTR1)/(2.0*STEFF)
      VECA2(3)=DEVIA(3)/STEFF
C*** CALCULATE VECTOR A3
      VECA3(1)=DEVIA(2)*DEVIA(4)+VARJ2/3.0
      VECA3(2)=DEVIA(1)*DEVIA(4)+VARJ2/3.0
      VECA3(3)=-2.0*DEVIA(3)*DEVIA(4)
      VECA3(4)=DEVIA(1)*DEVIA(2)-DEVIA(3)*DEVIA(3)+VARJ2/3.0
      GO TO (1,2,3,4) NCRIT
C*** TRESCA
    1 CONS1=0.0
      ABTHE=ABS(THETA*57.29577951308)
      IF(ABTHE.LT.29.0) GO TO 20
      CONS2=ROOT3
      CONS3=0.0
      GO TO 40
   20 CONS2=2.0*(COSTH+SINTH*TAN(3.0*THETA))
      CONS3=ROOT3*SINTH/(VARJ2*COST3)
      GO TO 40
C*** VON MISES
    2 CONS1=0.0
      CONS2=ROOT3
      CONS3=0.0
      GO TO 40
C*** MOHR-COULOMB
    3 CONS1=SIN(FRICT*0.017453292)/3.0
      ABTHE=ABS(THETA*57.29577951308)
      IF(ABTHE.LT.29.0) GO TO 30
      CONS3=0.0
      PLUMI=1.0
      IF(THETA.GT.0.0) PLUMI=-1.0
      CONS2=0.5*(ROOT3+PLUMI*CONS1/ROOT3)
      GO TO 40
   30 TANT3=TAN(3.0*THETA)
      CONS2=COSTH*((1.0+TANTH*TANT3)+CONS1*(TANT3-TANTH)/ROOT3)
      CONS3=(ROOT3*SINTH+CONS1*COSTH)/(2.0*VARJ2*COST3)
      GO TO 40
C*** DRUCKER-PRAGER
    4 SNPHI=SIN(FRICT*0.017453292)
      CONS1=2.0*SNPHI/(ROOT3*(3.0-SNPHI))
      CONS2=1.0
      CONS3=0.0
   40 CONTINUE
      DO 50 ISTR1=1,NSTR1
   50 AVECT(ISTR1)=CONS1*VECA1(ISTR1)+CONS2*
     .VECA2(ISTR1)+CONS3*VECA3(ISTR1)
      RETURN
      END
```

10.7 Examples

10.7.1 Introduction

To illustrate the use of DYNPAK we now describe the nonlinear transient dynamic analysis of (i) a spherical shell and (ii) a concrete gravity dam.

10.7.2 Spherical shell example

The shell,[8] shown in Fig. 10.3, is subjected to a distributed step pressure of 600 lb/in^2. The material is assumed to obey the Von Mises yield condition with linear isotropic hardening. The dimensions and properties of the shell are given as follows:

Internal radius	$R = 22.27$ in
Thickness of shell	$t = 0.41$ in
Semi angle	$\alpha = 26.67°$
Elastic modulus	$E = 10.5 \times 10^6$ lb/in^2
Poisson's ratio	$\nu = 0.3$
Yield stress	$\sigma_Y = 0.024 \times 10^6$ lb/in^2
Tangent hardening modulus	$E_T = 0.21 \times 10^6$ lb/^{2}
Mass density	$\rho = 2.45 \times 10^{-4}$ lb-sec^2/in^4
Step distributed pressure	$p = 600$ lb/in^2

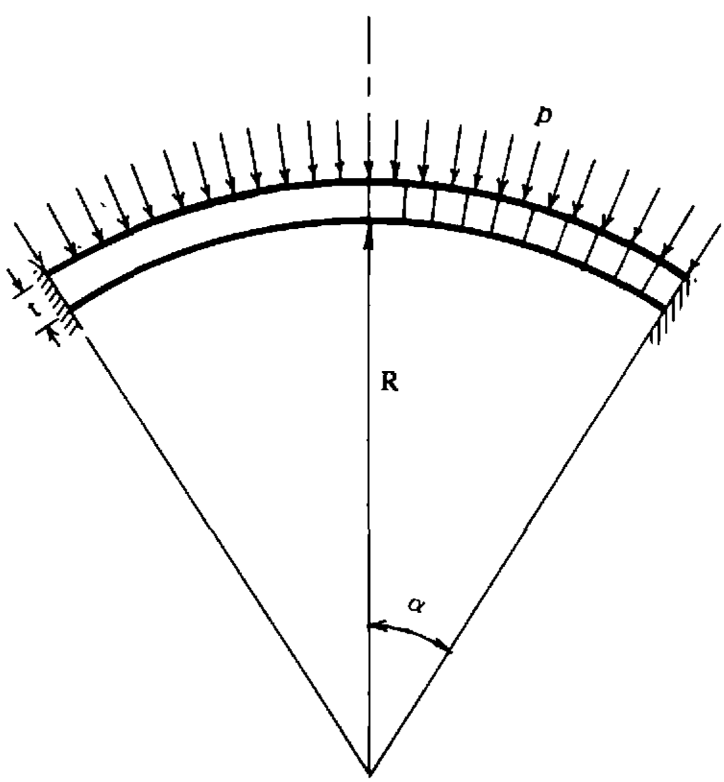


Fig. 10.3 Spherical shell and finite element mesh.

The shell is divided into ten, 8-noded, axisymmetric, isoparametric elements. The fundamental period of the shell is $T_f = 0.55 \times 10^{-3}$ sec, (Reference 8). For explicit central difference analysis, the time step is taken as 0.4×10^{-6} sec.

In order to illustrate the versatility of program DYNPAK we consider the following three cases:

(i) Small elastic displacements
(ii) Large elastic displacements
(iii) Small elasto-viscoplastic displacements (with a fluidity parameter value of $\gamma = 100.0$).

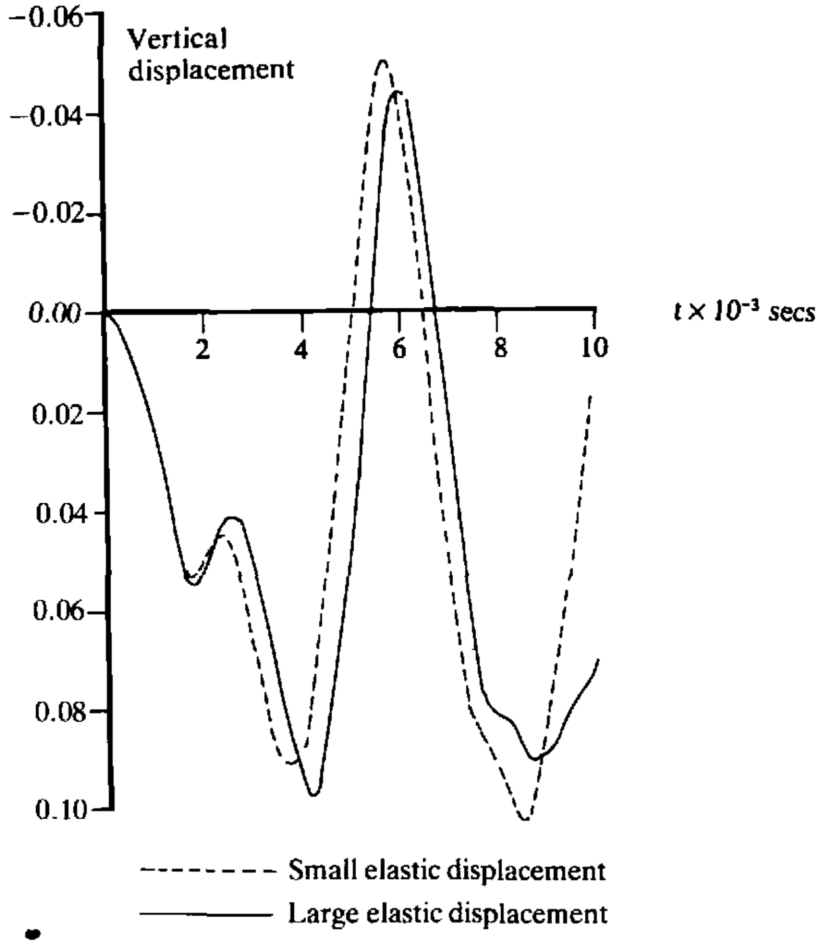


Fig. 10.4(a) Results of the transient dynamic analysis of a spherical shell cap. Cases (i) and (ii).

Figure 10.4(a) shows the vertical displacement of the crown lower point for the analyses based on both small and large elastic displacement assumptions. The results show that the inclusion of geometrically nonlinear effects in the analysis elongates the period. Figure 10.4(b) shows the small displacement, elasto-viscoplastic response (Case (iii)) of the spherical shell cap in which the value of the fluidity parameter is taken as $\gamma = 100.0$. It should be noted that permanent viscoplastic deflections occur thus providing a completely different response to either of the elastic responses shown in Fig. 10.4(a).

In Chapter 11 this problem is repeated using an elasto-plastic material model. It should be noted that in order to simulate elasto-plastic behaviour with DYNPAK a high value of the fluidity parameter (say $\gamma = 10000.0$)

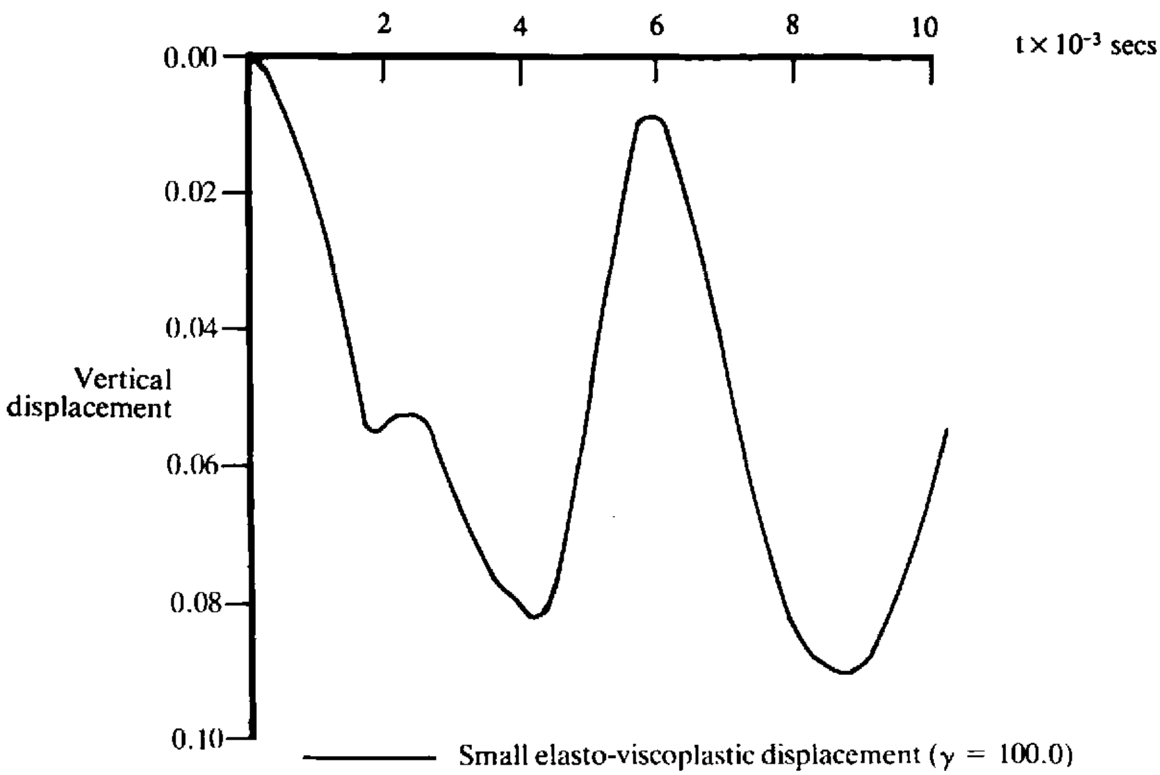


Fig. 10.4(b) Results of the transient dynamic analysis of a spherical shell cap. Case (iii).

should be adopted. Interested readers may like to compare DYNPAK and MIXDYN for elasto-plastic behaviour using a high fluidity parameter. However, care should be taken since the use of high fluidity parameter values requires the use of a smaller time step when an Euler scheme is used to evaluate the viscoplastic strains (see Section 8.3). Typical input data for Case (ii) are given in Appendix IV.

At this stage it is probably worth mentioning the important problem of combining material and geometric nonlinearities. Among the several papers on this topic in the existing literature we suggest that the interested reader could profitably refer to the following as a starting point for further study:

McMeeking, R. M. and Rice, J. R., Finite element formulations for problems of large elastic-plastic deformation, *Int. J. Solids Structures*, **11**, 601–616 (1975).

Hibbitt, H. D., Marcal, P. V. and Rice, J. R., A finite element formulation for problems of large strain and large displacement, *Int. J. Solids Structures*, **6**, 1069–1086 (1970).

Bathe, K. J., Ramm, E. and Wilson, E. L., Finite element formulations for large deformation analysis, *Int. J. Num. Meth. Engng.*, **9**, 353–386 (1975).

10.7.3 Gravity dam example

The geometry of the dam, the seismic acceleration history, the water level and material properties for both dam and foundation are arbitrary.

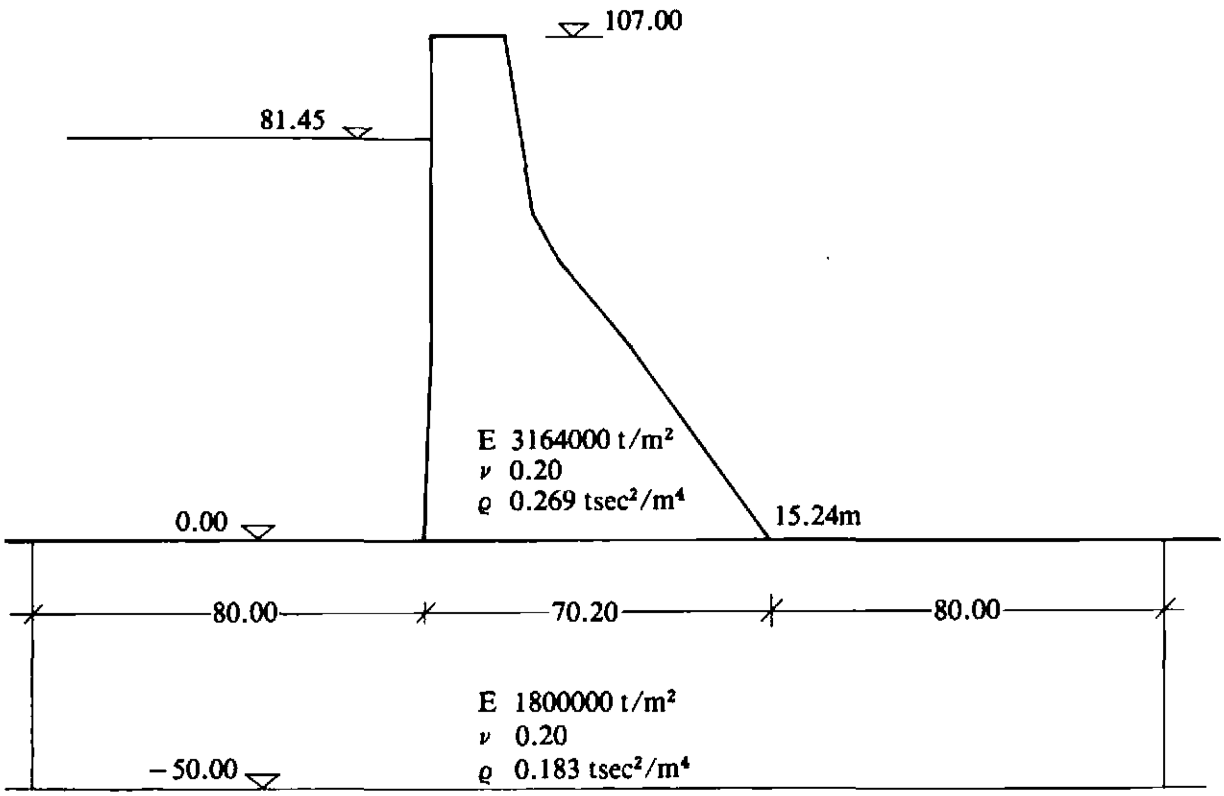


Fig. 10.5(a) Concrete gravity dam.

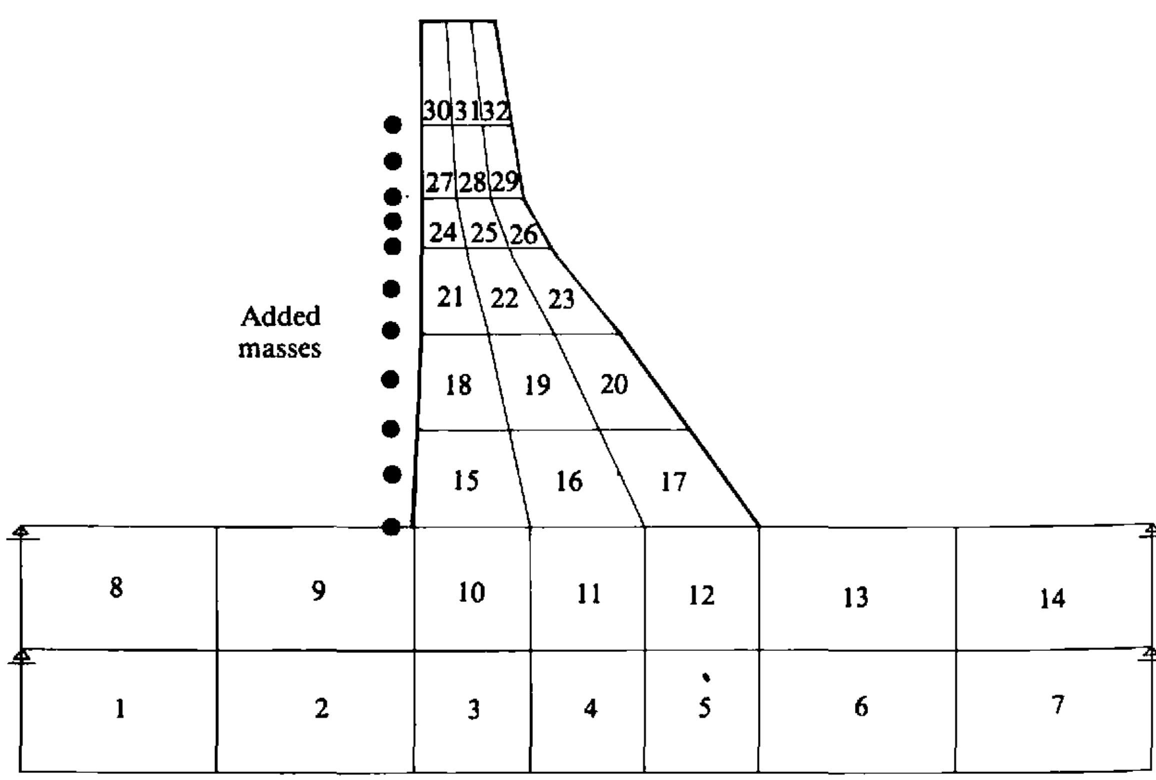


Fig. 10.5(b) Finite element mesh for concrete gravity dam.

Both the gravity dam and the foundation shown in Fig. 10.5(a) are idealized with two-dimensional, plane-strain, 8-noded isoparametric elements as shown in Fig. 10.5(b), using a 2×2 Gauss integration rule for the stiffness evaluation, and using a special mass lumping scheme with a 3×3 Gauss integration rule. The adopted 2×2 Gauss integration rule for the stiffness terms ensures that no locking behaviour will occur in the mesh, whereas the 3×3 Gauss integration rule for the lumped mass matrix terms renders better mass representation. The model base is assumed to be fixed, i.e. $u = v = 0$, and side boundaries are represented by horizontal rollers, i.e. $v = 0$.

A short duration analytic earthquake (sinesweep)[9] with a maximum acceleration level 0.33 g (developed as an equivalent to the El Centro NS accelerogram) will be used as a prescribed horizontal acceleration history at the model base level. It is assumed that this signal is the result of the deconvolution process of a prescribed signal at the foundation level. The displacements obtained in the solution process are relative to the model base.

Both the concrete and rock are assumed to behave as elasto-viscoplastic materials with no hardening. The Mohr–Coulomb yield surface is adopted, and the parameters c and ϕ are obtained from the uniaxial properties f_{cu} and f_t as indicated in Table 10.3.

$$f_t, f_{cu} = \text{tensile, compressive strengths of concrete,}$$

$$a = \frac{f_t}{f_{cu}} = \frac{1-\sin\phi}{1+\sin\phi},$$

$$\phi = \arcsin\left(\frac{1-a}{1+a}\right),$$

$$c = \frac{(a)^{-1/2}}{2} f_{cu},$$

$$F_0 \text{ (Mohr–Coulomb)} = c\cos\phi.$$

	f_{cu} (t/m^2)	f_t (t/m^2)	a	c (t/m^2)	ϕ	$F_0 = c\cos\phi$ (t/m^2)
concrete	4000	500	0.125	707.11	62.73	323.94
rock	3600	400	0.133	547.72	61.93	257.75

Table 10.3 Mohr–Coulomb yield surface parameters for concrete dam example.

The values of the fluidity parameters γ are considered to be the same for both the concrete and rock materials. Values of $\gamma = 0.00001$ and $\gamma = 0.001$ have been used for the two analyses presented. The stress level in the structure prior to the seismic excitation is assumed to be due to the self-weight and hydrostatic pressure of the water only.

The influence of the reservoir water on the dynamic behaviour of the dam is considered by taking into account the mass of water attached to the upstream face of the dam. The simple representation of 'added mass' with concentrated masses is used. The adopted model could be improved significantly with transmitting boundaries, better 'added mass' representation, a more realistic signal and a finer mesh.

The choice of the time step length depends on two criteria. For the explicit central difference integration scheme of the dynamic equilibrium equations, the highest mesh frequency defines the critical time step length

$$\Delta t_{CD} = \frac{2}{\omega_{\max}} \simeq \mu L\left(\frac{\rho(1+\nu)(1-2\nu)}{E(1-\nu)}\right)^{1/2}. \tag{10.51}$$

For the integration of the equations, which govern viscoplastic straining using the Euler method, the critical time step for the Mohr–Coulomb viscoplastic material is defined as

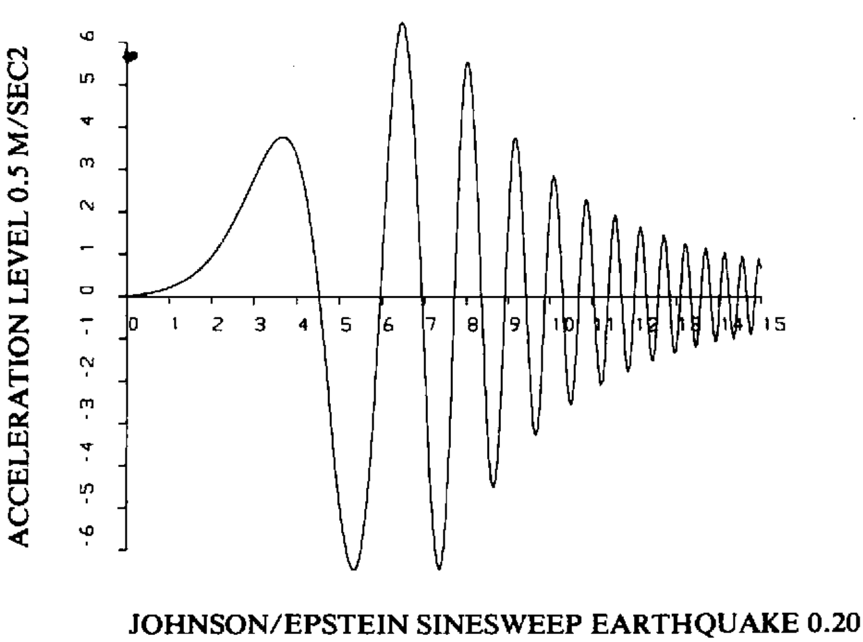
$$\Delta t_{MC} = \frac{4(1+\nu)(1-2\nu)\,c\cos\phi}{\gamma(1-2\nu+\sin^2\phi)}. \tag{10.52}$$

For the mathematical model under consideration, ($L = 2.4665$ m), the choice of the time step is governed by the Δt_{CD} criterion for both analyses. Note that since

$$\Delta t_{CD} = 0.000478 \text{ sec} \tag{10.53}$$

the adopted time step length is $\Delta t = 0.0004$ sec.

On the basis of the adopted mathematical model, (Fig. 10.5), input data can be prepared following the user notes, given in the Appendix III.



JOHNSON/EPSTEIN SINESWEEP EARTHQUAKE 0.20 SEC

Fig. 10.6(a) Johnson/Epstein sinesweep earthquake.

SINESWEEP DT 0.01 SEC 300 ENTRIES

0.0034	0.0069	0.0104	0.0140	0.0177	0.0215	0.0255
0.0296	0.0339	0.0385	0.0433	0.0484	0.0539	0.0597
0.0659	0.0725	0.0795	0.0871	0.0951	0.1038	0.1130
0.1229	0.1335	0.1449	0.1570	0.1700	0.1838	0.1986
0.2144	0.2312	0.2491	0.2681	0.2884	0.3098	0.3326
0.3567	0.3823	0.4092	0.4377	0.4677	0.4992	0.5324
0.5672	0.6036	0.6417	0.6815	0.7229	0.7660	0.8106
0.8568	0.9046	0.9537	1.0042	1.0558	1.1086	1.1622
1.2165	1.2713	1.3263	1.3812	1.4357	1.4894	1.5420
1.5930	1.6419	1.6881	1.7312	1.7705	1.8054	1.8351
1.8589	1.8761	1.8859	1.8874	1.8797	1.8621	1.8337
1.7935	1.7408	1.6747	1.5945	1.4993	1.3887	1.2621
1.1191	0.9594	0.7829	0.5899	0.3805	0.1554	-0.0845
-0.3381	-0.6038	-0.8798	-1.1638	-1.4533	-1.7372	-1.9899
-2.2286	-2.4500	-2.6507	-2.8273	-2.9764	-3.0948	-3.1793
-3.2271	-3.2356	-3.2025	-3.1262	-3.0056	-2.8402	-2.6303
-2.3768	-2.0819	-1.7485	-1.3804	-0.9825	-0.5607	-0.1220
0.3258	0.7742	1.2139	1.6349	2.0272	2.3806	2.6849
2.9306	3.1090	3.2125	3.2351	3.1726	3.0230	2.7867
2.4670	2.0698	1.6041	1.0818	0.5176	-0.0715	-0.6660
-1.2454	-1.7879	-2.2718	-2.6762	-2.9821	-3.1733	-3.2373
-3.1663	-2.9582	-2.6169	-2.1529	-1.5831	-0.9256	-0.2197
0.4796	1.1375	1.7207	2.1988	2.5461	2.7439	2.7810
2.6553	2.3743	1.9551	1.4235	0.8133	0.1640	-0.4813
-1.0789	-1.5873	-1.9703	-2.2002	-2.2599	-2.1450	-1.8645
-1.4408	-0.9084	-0.3116	0.2988	0.8699	1.3510	1.6985
1.8803	1.8793	1.6956	1.3476	0.8703	0.3129	-0.2659
-0.8041	-1.2427	-1.5330	-1.6417	-1.5565	-1.2875	-0.8674
-0.3482	0.2047	0.7201	1.1307	1.3817	1.4390	1.2948
0.9696	0.5104	-0.0151	-0.5283	-0.9507	-1.2170	-1.2848
-1.1436	-0.8166	-0.3588	0.1518	0.6265	0.9814	1.1528
1.1094	0.8596	0.4508	-0.0377	-0.5100	-0.8715	-1.0488
-1.0054	-0.7506	-0.3392	0.1389	0.5778	0.8784	0.9720
0.8371	0.5057	0.0580	-0.3962	-0.7437	-0.8966	-0.8159
-0.5228	-0.0956	0.3502	0.6922	0.8352	0.7389	0.4312
0.0025	-0.4198	-0.7084	-0.7749	-0.5989	-0.2364	0.1960
0.5575	0.7285	0.6519	0.3541	-0.0615	-0.4488	-0.6697
-0.6446	-0.3831	0.0171	0.4045	0.6304	0.6073	0.3446
-0.0521	-0.4214	-0.6111	-0.5422	-0.2444	0.1539	0.4789
0.5870	0.4302	0.0801	-0.3015	-0.5364	-0.5135	-0.2443
0.1398	0.4490	0.5290	0.3396	-0.0213	-0.3657	-0.5121
-0.3826	-0.0479	0.3081	0.4876	0.3899	0.0713	-0.2837
-0.4675	-0.3716	-0.0541	0.2916	0.4528	0.3295	

Fig. 10.6(b) Digital form of Johnson/Epstein sinesweep earthquake.

Prior to the dynamic analysis, the initial stresses σ_0 must be evaluated using some static finite element program. Nodal loads and the stress state for every Gauss integration point are recorded, and added to the input data for the dynamic analysis. The sinesweep accelerogram and 300 readings for $\Delta t = 0.01$ sec are given in Fig. 10.6. The accelerogram information is read in from a separate input unit (here tape 7, the assumed seismic excitation in the horizontal direction).

The displacement histories for selected nodal points and stress histories for selected Gauss integration points are written on separate output units (tape 10, tape 11) and may be used later for plotting the results. The displacement histories for nodal points 51 (structure base level) and 127 (dam crest) are given in Fig. 10.7.

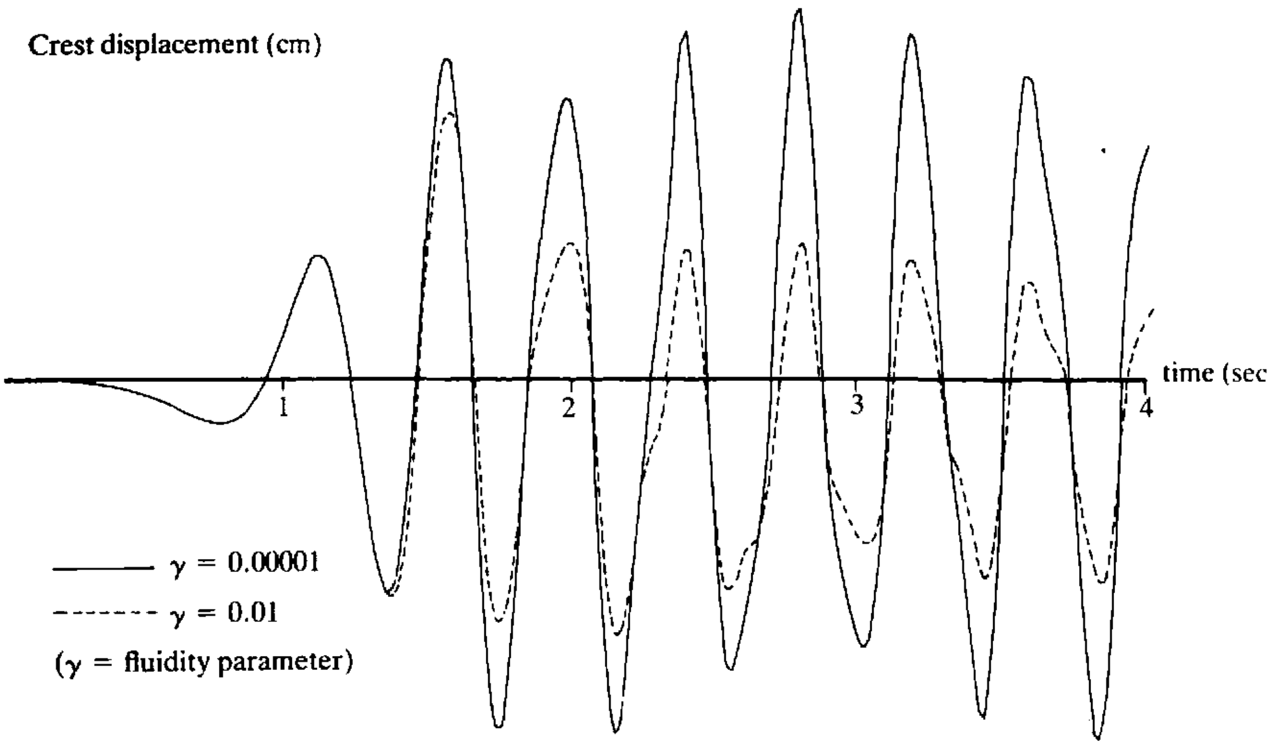


Fig. 10.7 Results of transient dynamic analysis of a concerte gravity dam.

10.8 Problems

10.1 A simply supported beam is subjected to a step uniformly distributed load. The dimensions and material properties of the beam are shown in Fig. 10.8(a). Only one quarter of the beam needs to be analysed as shown in Fig. 10.8(b). Use DYNPAK to find the midspan lateral deflection when the step lateral load is 0.75 p_0 where p_0 is the static collapse load. Note that this problem has been solved by Liu and Lin[10], Bathe *et al.*[11] and Nagarajan and Popov.[12] Use the Von Mises yield criterion, a high value of the fluidity parameter γ and 8-node elements.

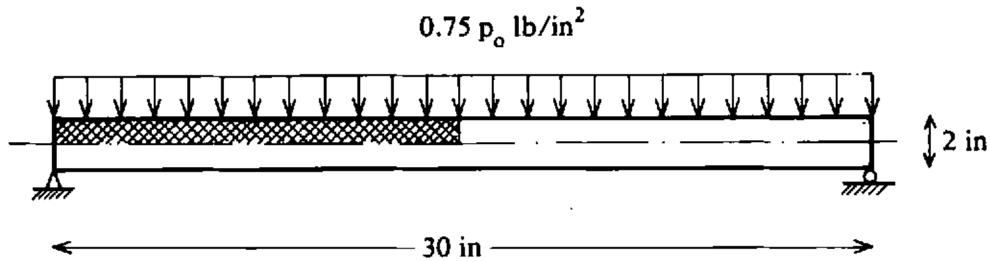
10.2 Repeat Problem 10.1 using the Tresca yield criterion.

10.3 Repeat Problem 10.1 using loads of intensity 0.625 p_0 and 0.50 p_0. Compare your results with those of Liu and Lin.[10]

10.4 For a step lateral load of 0.625 p_0, repeat Problem 10.1 for various degrees of hardening. Compare your results with those of Liu and Lin.[10]

10.5 Solve the problem given in Chapters 7 and 8 using dynamic relaxation.[13,14]

10.6 Implement an explicit elasto-plastic, transient dynamic, Mindlin plate program based on DYNPAK. Typical examples are given elsewhere.[15,16]



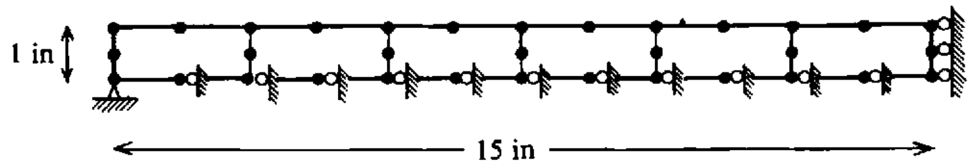


Fig. 10.8 Simply supported beam example (a) Geometry and loading, (b) Finite element idealisation.

10.9 References

1. Nickell, R. E., Nonlinear dynamics by mode superposition, *Comp. Meth. Appl. Mech. Engng*. **7**, 107–129 (1976).
2. Bathe, K. J. and Wilson, E. L., *Numerical Methods in Finite Element Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, 308–344 (1976).
3. Nayak, G. C., Plasticity and large deformation by the finite element method, Ph.D. Thesis, University College of Swansea (1971).
4. Irons, B. M. and Ahmad, S., *Finite Element Techniques*, Ellis Horwood, Chichester (1980).
5. Bićanić, N., Nonlinear finite element transient response of concrete structures, Ph.D. Thesis, University College of Swansea (1978).
6. Hinton, E., Rock, T. A. and Zienkiewicz, O. C., A note on mass lumping and related processes in the finite element method, *Int. J. Earthquake Engng. Struct. Dynamics*, **4**, 246–249 (1976).
7. Hinton, E. and Owen, D. R. J., *Finite Element Programming*, Academic Press, London (1977).
8. Bathe, K. J. and Ozdemir, H., Elasto-plastic large deformation static and dynamic analysis, *Computers and Structures*, **6**, 81–92 (1976).
9. Johnson, G. R. and Epstein, H. I., Short duration analytic earthquake, *J. ASCE, Struct. Div.*, **102**, No. ST5, 993–1000 (1976).
10. Liu, S. C. and Lin, T. H., Elastic-plastic dynamic analysis of structures using known elastic solutions, *Int. J. Earthquake Engng. Struct. Dynamics*, **7**, 147–159 (1979).
11. Bathe, K. J., Ozdemir, H. and Wilson, E. L., Static and geometric and material nonlinear analysis, *Structures and Materials Research Report* No. UC SESM 74-4, University of California, Berkeley (1974).
12. Nagarajan, S. and Popov, E. P., Elastic-plastic dynamic analysis of axisymmetric solid, *Computers and Structures*, **4**, 1117–1134 (1974).
13. Pica, A. and Hinton, E., Transient and pseudo-transient analysis of Mindlin plates, *Int. J. Num. Meth. Engng*. **15**, 189–208 (1980).
14. Brew, J. S. and Brotton, D. M., Nonlinear structural analysis by dynamic relaxation, *Int. J. Num. Meth. Engng*. **3**, 463–483 (1971).

15. HINTON, E., OWEN, D. R. J. and SHANTARAM, D., Dynamic transient non-linear behaviour of thick and thin plates, In: *The Mathematics of Finite Elements and Applications, MAFELAP II*, 1975, Ed. J. R. Whiteman, Academic Press, London, 423–438 (1977).
16. RAO, S. S. and RAGHAVAN, K. S., Dynamic response of inelastic thick plates, *AIAA J.*, **17**, 85–90 (1979).

Chapter 11
Implicit-explicit transient dynamic analysis

Written in collaboration with D. K. Paul

11.1 Introduction

In Chapter 10 we have shown that the explicit, central difference time stepping scheme is a simple and powerful method of time integration. The main drawback of the scheme is that it is conditionally stable. Thus the computational advantages of the central difference scheme are counter-balanced by the very small size of time step necessary when some stiff (and/or small) elements are present. For such problems the unconditionally stable implicit schemes permit the use of larger time steps, the size of which is governed only by accuracy considerations. Unfortunately these schemes which require matrix factorisations involve larger computer core storage and more operations per time step than the central difference scheme. The selection of a suitable time integration scheme is therefore largely a matter of experience.

In some problems, typified by the one illustrated in Fig. 11.1, we may be confronted with a situation in which there is a 'soft' subregion Ω^E where an

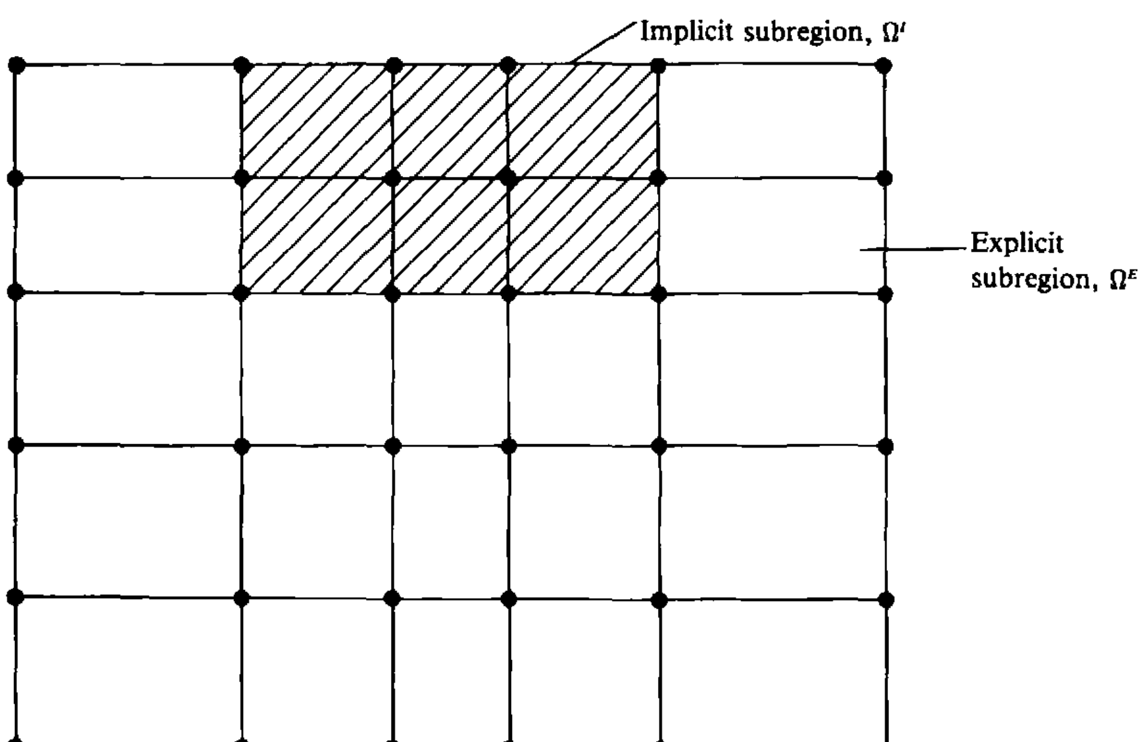


Fig. 11.1 Implicit-explicit partitioning.

explicit scheme is desirable and a 'stiff' subregion Ω^I where an implicit scheme is preferable for greater efficiency. In such cases it is possible to simultaneously make use of both implicit and explicit algorithms. Implicit–explicit schemes offer a unified approach to problems of structural transient dynamics and can lead to significant computational advantages.

Implicit–explicit schemes were first introduced by Belytschko and Mullen[1–3] and were given an alternative form by Hughes and co-workers[4–6] and Park *et al.*[7–8] It can be shown that the stability of such schemes is governed by the explicit elements.

In this chapter Implicit and Implicit–Explicit methods for nonlinear transient dynamic analysis are discussed and we follow the element partitioning approach described by Hughes. A program, named MIXDYN, for Implicit–Explicit linear and nonlinear transient dynamic analysis is included. Some numerical examples are solved to show some of the capabilities of the program. The same program could be modified for static analysis by some simple changes.

11.2 Implicit time integration

11.2.1 Newmark's algorithm

In order to introduce the implicit/explicit algorithm we describe the predictor-corrector form of the Newmark scheme for the integration of the semi-discrete system of equations which govern nonlinear transient dynamic problems. Typically at time station $t_n+\Delta t$ these equations take the form

$$\boldsymbol{M}\boldsymbol{a}_{n+1}+\boldsymbol{p}_{n+1}=\boldsymbol{f}_{n+1} \tag{11.1}$$

where $\boldsymbol{M}$, $\boldsymbol{a}_{n+1}$, $\boldsymbol{p}_{n+1}$ and $\boldsymbol{f}_{n+1}$ are the mass matrix, acceleration vector, internal force vector (which may depend on the displacements $\boldsymbol{d}_{n+1}$ and velocities $\dot{\boldsymbol{d}}_{n+1}$ and their histories) and applied force vector respectively. Let

$$[\boldsymbol{K}_T]_{n+1} = \partial\boldsymbol{p}_{n+1}/\partial\boldsymbol{d}_{n+1} \text{ and } [\boldsymbol{C}_T]_{n+1} = \partial\boldsymbol{p}_{n+1}/\partial\dot{\boldsymbol{d}}_{n+1} \tag{11.2}$$

denote the tangent stiffness and damping matrices respectively.

In the Newmark scheme we endeavour to satisfy the following equations

$$\boldsymbol{M}\boldsymbol{a}_{n+1}+\boldsymbol{p}_{n+1}=\boldsymbol{f}_{n+1} \tag{11.3}$$

$$\boldsymbol{d}_{n+1}=\tilde{\boldsymbol{d}}_{n+1}+\Delta t^2\beta\boldsymbol{a}_{n+1} \tag{11.4}$$

$$v_{n+1} = \tilde{v}_{n+1} + \Delta t \gamma a_{n+1} \tag{11.5}^*$$

where

$$\tilde{d}_{n+1} = d_n + \Delta t v_n + \Delta t^2 (1-2\beta) a_n / 2 \tag{11.6}$$

$$\tilde{v}_{n+1} = v_n + \Delta t (1-\gamma) a_n. \tag{11.7}$$

Note that d_n, v_n and a_n are the approximations to $d(t_n)$, $\dot{d}(t_n)$ and $\ddot{d}(t_n)$ and β and γ are free parameters which control the accuracy and stability of the method. The values $\tilde{d}_{n+1}$ and $\tilde{v}_{n+1}$ are predictor values and d_{n+1} and v_{n+1} are corrector values.

Initially the displacements d_0 and velocities v_0 are provided and we find the accelerations a_0 from the expression

$$M a_0 = f_0 - p(d_0, v_0). \tag{11.8}$$

Thus a_0 may be found by a factorization, forward reduction and back substitution unless M is diagonal in which case the solution is trivial.

We then solve (11.3) to (11.7) by forming an 'effective static problem'† which is solved using a Newton Raphson type scheme, as described earlier. The algorithm is summarised in Table 11.1.

Table 11.1 Newmark's algorithm

1 Set iteration counter $i = 0$.

2 Begin predictor phase in which we set

$$d_{n+1}^{[i]} = \tilde{d}_{n+1} = d_n + \Delta t v_n + \Delta t^2(1-2\beta) a_n/2 \tag{i}$$
$$v_{n+1}^{[i]} = \tilde{v}_{n+1} = v_n + \Delta t(1-\gamma) a_n \tag{ii}$$
$$a_{n+1}^{[i]} = [d_{n+1}^{[i]} - \tilde{d}_{n+1}]/(\Delta t^2 \beta) = 0. \tag{iii}$$

3 Evaluate residual forces using the equation

$$\psi^{[i]} = f_{n+1} - M a_{n+1}^{[i]} - p(d_{n+1}^{[i]}, v_{n+1}^{[i]}). \tag{iv}$$

4 If required, form the effective stiffness matrix using the expression

$$K^* = M/(\Delta t^2 \beta) + \gamma C_T/(\Delta t \beta) + K_T(d_{n+1}^{[i]}). \tag{v}$$

Otherwise use a previously calculated K^*.

5 Factorize, forward reduction and backsubstitute as required to solve

$$K^* \Delta d^{[i]} = \psi^{[i]}. \tag{vi}$$

6 Enter corrector phase in which we set

$$d_{n+1}^{[i+1]} = d_{n+1}^{[i]} + \Delta d^{[i]} \tag{vii}$$
$$a_{n+1}^{[i+1]} = [d_{n+1}^{[i+1]} - \tilde{d}_{n+1}]/(\Delta t^2 \beta) \tag{viii}$$
$$v_{n+1}^{[i+1]} = v_{n+1} + \Delta t \gamma a_{n+1}^{[i+1]}. \tag{ix}$$

7 If $\Delta d^{[i]}$ and/or $\psi^{[i]}$ do not satisfy the convergence conditions then set $i = i+1$ and go to step 3, otherwise continue.

8 Set

$$d_{n+1} = d_{n+1}^{[i+1]} \tag{x}$$
$$v_{n+1} = v_{n+1}^{[i+1]} \tag{xi}$$
$$a_{n+1} = a_{n+1}^{[i+1]} \tag{xii}$$

for use in the next time step. Also set $n = n+1$, form p and begin next time step.

---

* In this chapter γ is a Newmark parameter and not the viscoplastic fluidity parameter.

† $K^* \Delta d^{[i]} = \psi^{[i]}$.

11.2.2 Predictor-corrector algorithm

Let us now consider an 'explicit' algorithm associated with the Newmark schemes described earlier. In this explicit predictor-corrector algorithm we assume that the mass matrix $\boldsymbol{M}$ is diagonal and we make use of the expression

$$\boldsymbol{M}\boldsymbol{a}_{n+1}+\boldsymbol{p}(\tilde{\boldsymbol{d}}_{n+1},\ \tilde{\boldsymbol{v}}_{n+1}) = \boldsymbol{f}_{n+1} \tag{11.9}$$

Notice that the calculation is explicit since we use corrector values obtained from information given in the previous step.

As we would like to eventually combine the implicit and explicit methods we organise our implementation of this explicit method in a similar fashion to the implementation given of the implicit scheme in the previous section. Table 11.2 summarises the algorithm.

Table 11.2 Explicit predictor-corrector algorithm

1	Begin predictor phase by setting	
	$\boldsymbol{d}_{n+1}{}^{[0]} = \tilde{\boldsymbol{d}}_{n+1} = \boldsymbol{d}_n+\Delta t\boldsymbol{v}_n+\Delta t^2(1-2\beta)\boldsymbol{a}_n/2$	(i)
	$\boldsymbol{v}_{n+1}{}^{[0]} = \tilde{\boldsymbol{v}}_{n+1} = \boldsymbol{v}_n+\Delta t(1-\gamma)\boldsymbol{a}_n$	(ii)
	$\boldsymbol{a}_{n+1}{}^{[0]} = \boldsymbol{0}.$	(iii)
2	Evaluate the residual forces using the equation	
	$\boldsymbol{\psi}^{[0]} = \boldsymbol{f}_{n+1}-\boldsymbol{p}(\boldsymbol{d}_{n+1}{}^{[0]},\ \boldsymbol{v}_{n+1}{}^{[0]}).$	(iv)
3	If required, form the 'effective' stiffness matrix using the expression	
	$\boldsymbol{K}^* = \boldsymbol{M}/(\Delta t^2\beta).$	(v)
	Note that as the mass matrix $\boldsymbol{M}$ does not change $\boldsymbol{K}^*$ will be formed once only.	
4	Perform factorization, forward reduction and backsubstitution as required to solve	
	$\boldsymbol{K}^*\Delta\boldsymbol{d}^{[0]} = \boldsymbol{\psi}^{[0]}$	(vi)
5	Enter the corrector phase in which we set	
	$\boldsymbol{d}_{n+1}{}^{[1]} = \boldsymbol{d}_{n+1}{}^{[0]}+\Delta\boldsymbol{d}^{[0]}$	(vii)
	$\boldsymbol{a}_{n+1}{}^{[1]} = [\boldsymbol{d}_{n+1}{}^{[1]}-\tilde{\boldsymbol{d}}_{n+1}]/(\Delta t^2\beta)$	(viii)
	$\boldsymbol{v}_{n+1}{}^{[1]} = \boldsymbol{v}_{n+1}+\Delta t\gamma\boldsymbol{a}_{n+1}{}^{[1]}.$	(ix)
6	Set $\boldsymbol{d}_{n+1} = \boldsymbol{d}_{n+1}{}^{[1]}$	(x)
	$\boldsymbol{v}_{n+1} = \boldsymbol{v}_{n+1}{}^{[1]}$	(xi)
	$\boldsymbol{a}_{n+1} = \boldsymbol{a}_{n+1}{}^{[1]}$	(xii)
	for use in the next time step. Also set $n = n+1$, form $\boldsymbol{p}$ and begin next time step.	

11.3 Implicit-explicit algorithm

11.3.1 Introduction

We now combine the methods described in Sections 11.2.1 and 11.2.2 so that the finite element mesh contains two groups of elements: the implicit group and the explicit group. The superscripts I and E will henceforth refer to the implicit and explicit groups respectively.

In the implicit–explicit algorithm we iterate within each time step in order to satisfy the equation

$$\boldsymbol{M}\boldsymbol{a}_{n+1}+\boldsymbol{p}^I(\boldsymbol{d}_{n+1},\boldsymbol{v}_{n+1})+\boldsymbol{p}^E(\tilde{\boldsymbol{d}}_{n+1},\tilde{\boldsymbol{v}}_{n+1}) = \boldsymbol{f}_{n+1} \qquad (11.10)$$

in which $\boldsymbol{M} = \boldsymbol{M}^I+\boldsymbol{M}^E$ and $\boldsymbol{f}_{n+1} = \boldsymbol{f}_{n+1}{}^I+\boldsymbol{f}_{n+1}{}^E$. Note that we assume $\boldsymbol{M}^E$ is diagonal.

11.3.2 The structure of the effective stiffness matrix

The algorithm, which is summarised in Table 11.3, is very similar to the implicit algorithm given in Section 11.2.2. The profile structure of $\boldsymbol{K}^*$ is very interesting. It has diagonal subregions corresponding to the explicit group of elements. Elsewhere, $\boldsymbol{K}^*$ has a profile structure which corresponds to the connectivity of the implicit group only.

Table 11.3 Implicit–explicit algorithm

1	Set iteration counter $i = 0$.	
2	Begin predictor phase in which we set	
	$\boldsymbol{d}_{n+1}{}^{[i]} = \tilde{\boldsymbol{d}}_{n+1} = \boldsymbol{d}_n+\Delta t\boldsymbol{v}_n+\Delta t^2(1-2\beta)\boldsymbol{a}_n/2$	(i)
	$\boldsymbol{v}_{n+1}{}^{[i]} = \tilde{\boldsymbol{v}}_{n+1} = \boldsymbol{v}_n+\Delta t(1-\gamma)\boldsymbol{a}_n$	(ii)
	$\boldsymbol{a}_{n+1}{}^{[i]} = [\boldsymbol{d}_{n+1}{}^{[i]}-\boldsymbol{d}_{n+1}]/(\Delta t^2\beta) = 0.$	(iii)
3	Evaluate residual forces using the equation	
	$\boldsymbol{\psi}^{[i]} = \boldsymbol{f}_{n+1}-\boldsymbol{M}\boldsymbol{a}_{n+1}{}^{[i]}-\boldsymbol{p}^I(\boldsymbol{d}_{n+1}{}^{[i]}, \boldsymbol{v}_{n+1}{}^{[i]})-\boldsymbol{p}^E(\tilde{\boldsymbol{d}}_{n+1}, \tilde{\boldsymbol{v}}_{n+1}).$	(iv)
4	If required, form the effective stiffness matrix using the expression	
	$\boldsymbol{K}^* = \boldsymbol{M}/(\Delta t^2\beta)+\gamma\boldsymbol{C}_T{}^I/(\Delta t\beta)+\boldsymbol{K}_T{}^I(\boldsymbol{d}_{n+1}{}^{[i]}).$	(v)
	Otherwise use a previously calculated $\boldsymbol{K}^*$.	
	(Note that $\boldsymbol{K}_T{}^I = \partial\boldsymbol{p}^I/\partial\boldsymbol{d}$ and $\boldsymbol{C}_T{}^I = \partial\boldsymbol{p}^I/\partial\boldsymbol{v}$).	
5	Perform factorization, forward reduction and backsubstitution as required to solve	
	$\boldsymbol{K}^*\Delta\boldsymbol{d}^{[i]} = \boldsymbol{\psi}^{[i]}.$	(vi)
6	Enter corrector phase in which we set	
	$\boldsymbol{d}_{n+1}{}^{[i+1]} = \boldsymbol{d}_{n+1}{}^{[i]}+\Delta\boldsymbol{d}^{[i]}$	(vii)
	$\boldsymbol{a}_{n+1}{}^{[i+1]} = [\boldsymbol{d}_{n+1}{}^{[i+1]}-\tilde{\boldsymbol{d}}_{n+1}]/(\Delta t^2\beta)$	(viii)
	$\boldsymbol{v}_{n+1}{}^{[i+1]} = \boldsymbol{v}_{n+1}+\Delta t\gamma\boldsymbol{a}_{n+1}{}^{[i+1]}.$	(ix)
7	If $\Delta\boldsymbol{d}^{[i]}$ and/or $\boldsymbol{\psi}^{[i]}$ do not satisfy the convergence conditions, then set $i = i+1$ and go to step 3, otherwise continue.	
8	Set $\boldsymbol{d}_{n+1} = \boldsymbol{d}_{n+1}{}^{[i+1]}$	(x)
	$\boldsymbol{v}_{n+1} = \boldsymbol{v}_{n+1}{}^{[i+1]}$	(xi)
	$\boldsymbol{a}_{n+1} = \boldsymbol{a}_{n+1}{}^{[i+1]}$	(xii)
	for use in the next time step. Also set $n = n+1$, form $\boldsymbol{p}$ and begin next time step.	

Consider the three meshes and effective stiffness matrices shown in Fig. 11.2(a)–(c):

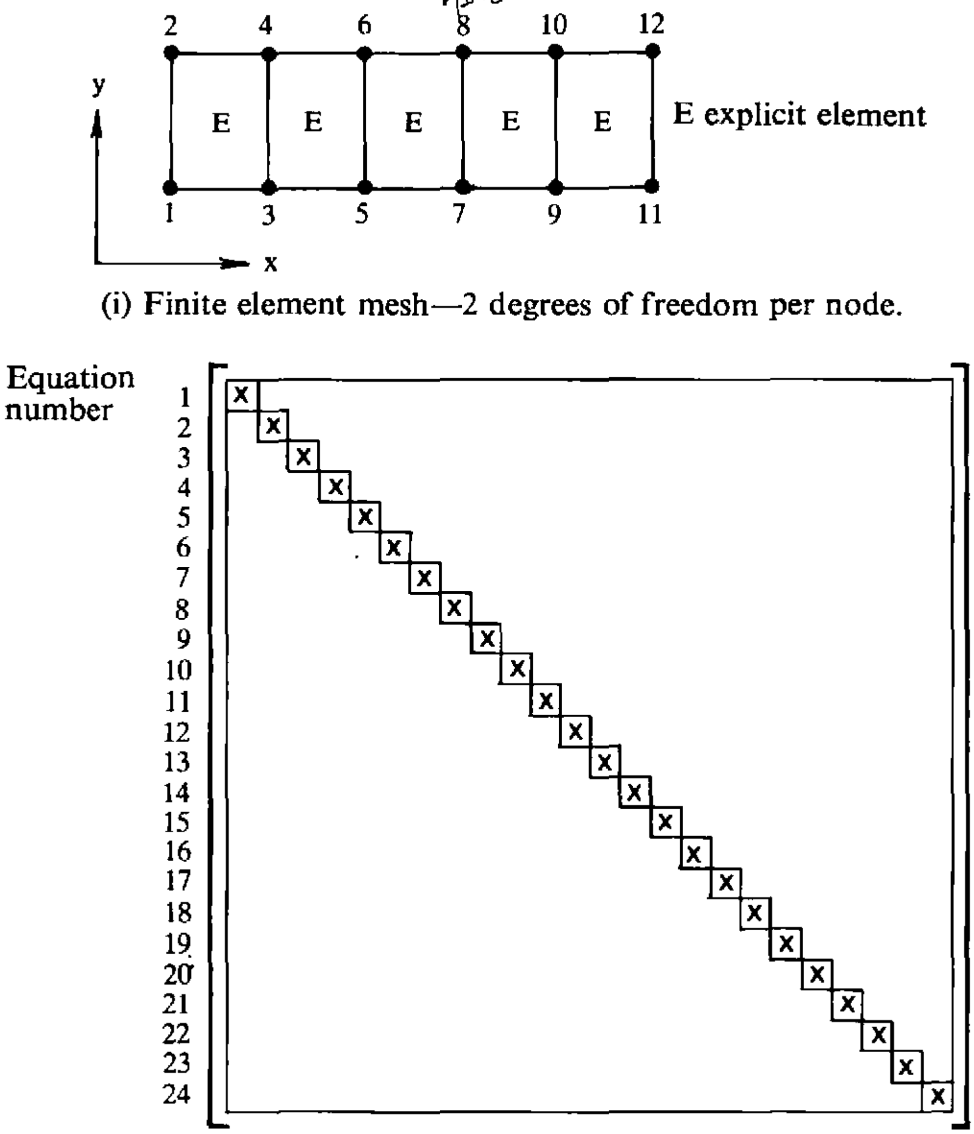
(i) When there are only explicit elements, $\boldsymbol{K}^*$ is diagonal. In other words $\boldsymbol{K}^*$ has the same profile structure as $\boldsymbol{M}^E$ (Fig. 11.2(a)).

(ii) For a mesh consisting of only implicit elements $\boldsymbol{K}^*$ has the same profile structure as $\boldsymbol{K}^I$ (Fig. 11.2(b)).

(iii) For the partitioned mesh containing both implicit and explicit groups we see the appropriate combination of parts of both profile structures (Fig. 11.2(c)).

To fully exploit the profile structure of K^*, Hughes *et al.*[4] have suggested the use of profile solvers. In our implementation of the scheme we adopt a slightly modified version of the in-core profile solver given by Bathe and Wilson.[9]

堆芯内



(i) Finite element mesh—2 degrees of freedom per node.

(ii) Profile of K^*.

Fig. 11.2(a) Two-dimensional finite element mesh and profile structure of the effective stiffness matrix K^* (explicit elements only).

11.3.3 Alternative predictor values

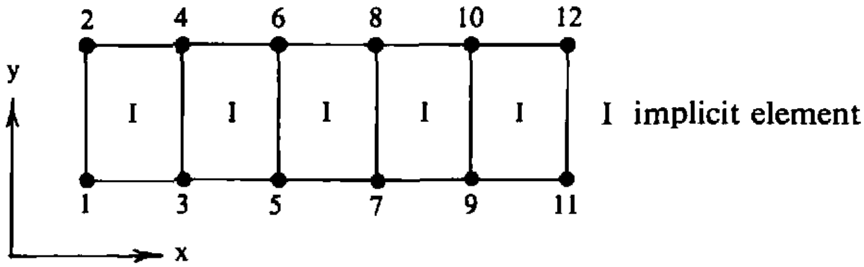
In equations (i)–(iii) in Table 11.3 we gave the approach described by Hughes and Liu.[4] For implicit–explicit problems other predictor values may be adopted. Here we consider two cases:

1. *Hughes and Liu predictor values*

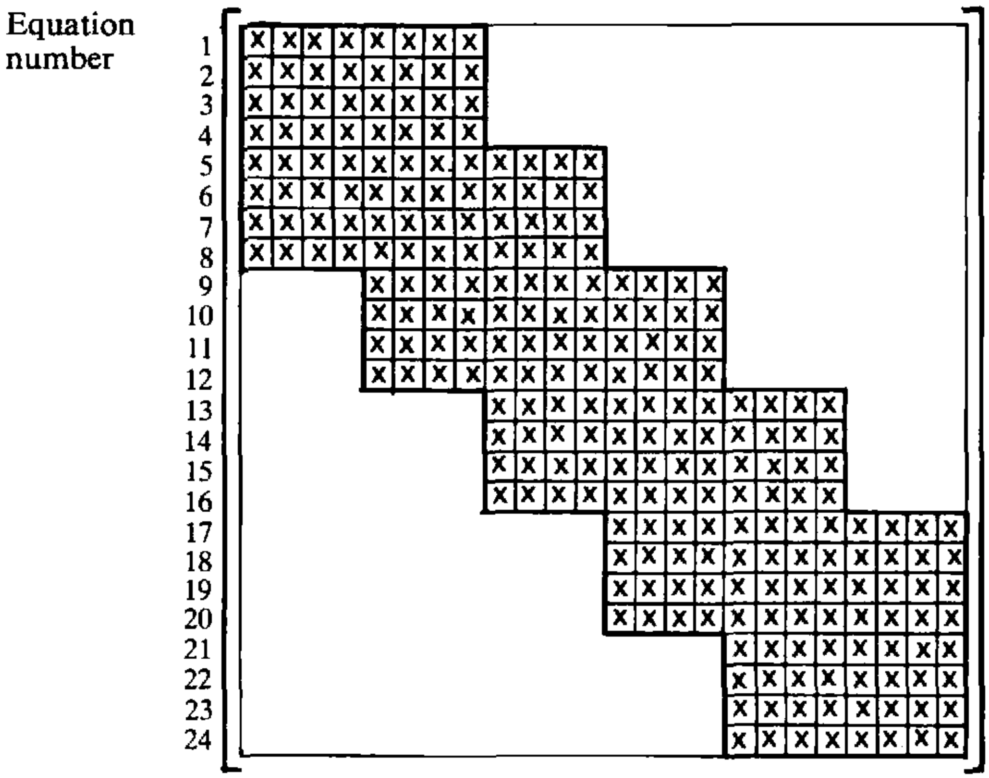
$$\boldsymbol{d}_{n+1}^{[0]} = \tilde{\boldsymbol{d}}_{n+1} = \boldsymbol{d}_n + \Delta t \boldsymbol{v}_n + \Delta t^2 (1-2\beta)\boldsymbol{a}_n/2 \qquad \text{(i)}$$

$$\boldsymbol{v}_{n+1}^{[0]} = \tilde{\boldsymbol{v}}_{n+1} = \boldsymbol{v}_n + \Delta t(1-\gamma)\boldsymbol{a}_n \qquad \text{(ii)}$$

$$\boldsymbol{a}_{n+1}^{[0]} = [\boldsymbol{d}_{n+1}^{[0]} - \tilde{\boldsymbol{d}}_{n+1}]/(\Delta t^2 \beta) \qquad \text{(iii)} \qquad (11.11)$$



(i) Finite element mesh—2 degrees of freedom per node.



(ii) Profile of K^*.

Fig. 11.2(b) Two-dimensional finite element mesh and profile structure of the effective stiffness matrix K^* (implicit elements only).

2. *Alternative predictor values*

$$d_{n+1}^{[0]} = d_n \qquad \text{(i)}$$

$$v_{n+1}^{[0]} = v_n \qquad \text{(ii)}$$

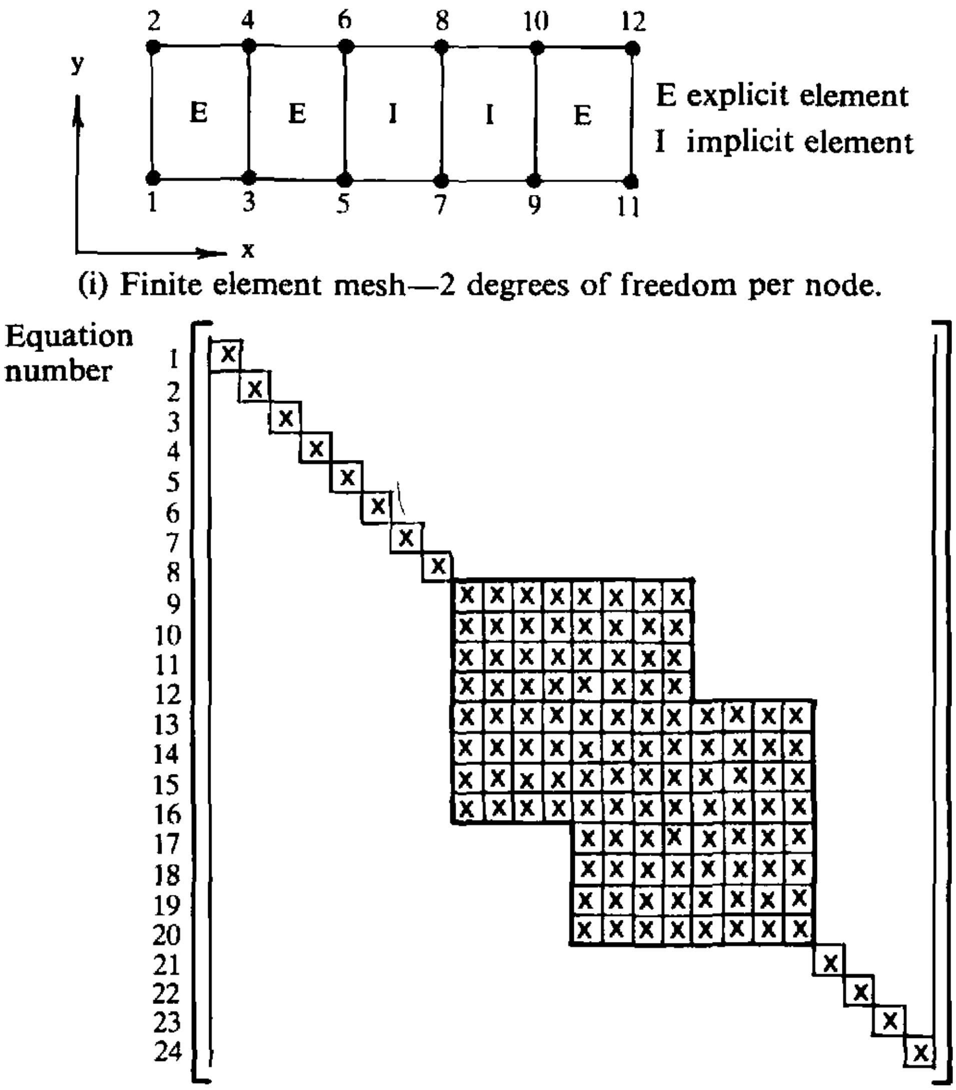
$$a_{n+1}^{[0]} = [d_{n+1}^{[0]} - \tilde{d}_{n+1}]/(\Delta t^2 \beta) \qquad \text{(iii)}$$

$$(\text{where } \tilde{d}_{n+1} = d_n + \Delta t v_n + \Delta t^2 (1-2\beta) a_n)/2 \qquad (11.12)$$

The second approach is recommended for elastoplastic problems for use with meshes involving only implicit elements in which $\gamma = \frac{1}{2}$ and when large time steps are adopted.

11.3.4 Stability limits

Hughes *et al.*[4] have discussed the stability limits for this implicit–explicit scheme.



(i) Finite element mesh—2 degrees of freedom per node.

(ii) Profile of K^*.

Fig. 11.2(c) Two-dimensional finite element mesh and profile structure of the effective stiffness matrix K^* (Implicit and explicit elements).

If $\gamma \geqslant \frac{1}{2}$ and $\beta = (\gamma+\frac{1}{2})^2/4$, we achieve unconditional stability in the implicit element group. The time step is then restricted by the explicit element group. For the case in which $\gamma = \frac{1}{2}$, the critical time step may be written as

$$\Delta t_{\text{crit}} = 2/\omega_{\max} \tag{11.13}$$

where $\omega_{\max}$ is the maximum frequency of the explicit group. We can estimate $\omega_{\max}$ as

$$\omega_{\max} \leqslant \max_{e}(\omega_{\max}{}^{(e)}) \tag{11.14}$$

where $\omega_{\max}{}^{(e)}$ is the maximum frequency of the e^{th} element of the explicit group.

Since K_T is changing from step to step, strictly speaking the maximum frequency should be estimated at the beginning of every step. In elasto-plastic analysis, the structure generally becomes more flexible and (11.14) may be used. However, for a better estimate of the critical time step the nonlinear eigenvalues should be evaluated.

If only implicit elements are used and if $\gamma \geq \frac{1}{2}$ and $\beta = (\gamma + \frac{1}{2})^2/4$, then error investigations carried out in terms of period elongation and amplitude decay with the increase of time step indicate that for reasonable accuracy the time step should be limited to 1/100 of the fundamental (largest) period. It is observed that the amplitude decay caused by the numerical integration errors effectively filters the higher mode response out of the solution in the Houbolt and Wilson θ method. However when we employ the Newmark constant–average–acceleration scheme, which does not introduce amplitude decay, the higher frequency response is retained in the solution. In order to obtain amplitude decay using the Newmark method, it is necessary to employ $\gamma > \frac{1}{2}$.

11.4 Evaluation of the tangential stiffness matrix

In program MIXDYN we adopt an elasto-plastic material model and therefore the stresses and the tangential stiffness matrix at any time station $t_n + \Delta t$ may be evaluated in the manner outlined in Chapter 7 for static problems. As an alternative geometrically nonlinear elastic effects are considered using a total Lagrangian formulation.

The internal resisting force vector for the implicit elements at time station $t_n + \Delta t$ is given as

$$\boldsymbol{p}_{n+1}{}^{I} = \int_{\Omega^I} [\boldsymbol{B}^I]^T{}_{n+1}\, \boldsymbol{\sigma}_{n+1}\, d\Omega \tag{11.15}$$

and therefore the tangential stiffness matrix may be written as

$$\frac{\partial \boldsymbol{p}_{n+1}{}^{I}}{\partial \boldsymbol{d}_{n+1}} = [\boldsymbol{K}_T{}^I]_{n+1} = \int_{\Omega^I} [\boldsymbol{B}^I]^T{}_{n+1}\, \boldsymbol{D}_{n+1} [\boldsymbol{B}^I]_{n+1}\, d\Omega + \int_{\Omega^I} [\boldsymbol{G}]^T{}_{n+1}\, \boldsymbol{S}_{n+1}\, \boldsymbol{G}_{n+1}\, d\Omega \qquad (11.16)^*$$

in which $\boldsymbol{D}_{n+1}$ is the elasto-plastic modulus matrix defined in Chapter 7, $[\boldsymbol{B}_{NL}{}^I]_{n+1}$ is the nonlinear strain-displacement matrix defined in Chapter 10, the matrix $\boldsymbol{S}_{n+1}$ is given as

$$\boldsymbol{S}_{n+1} = \begin{bmatrix} \sigma_x \boldsymbol{I}_2 & \tau_{xy} \boldsymbol{I}_2 \\ \tau_{xy} \boldsymbol{I}_2 & \sigma_y \boldsymbol{I}_2 \end{bmatrix}_{n+1} \tag{11.17}$$

for plane stress and plane strain problems, and

$$\boldsymbol{S}_{n+1} = \begin{bmatrix} \sigma_r \boldsymbol{I}_2 & \tau_{rz} \boldsymbol{I}_2 & \boldsymbol{0} \\ \tau_{rz} \boldsymbol{I}_2 & \sigma_z \boldsymbol{I}_2 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \sigma_\theta \end{bmatrix}_{n+1} \tag{11.18}$$

* The second matrix is only included for geometrically nonlinear problems.

for axisymmetric problems, and

$$[G_i]_{n+1} = \begin{bmatrix} \dfrac{\partial N_i}{\partial x} & 0 & \dfrac{\partial N_i}{\partial y} & 0 \\ 0 & \dfrac{\partial N_i}{\partial x} & 0 & \dfrac{\partial N_i}{\partial y} \end{bmatrix}^T \tag{11.19}$$

for plane stress and plane strain problems, and

$$[G_i]_{n+1} = \begin{bmatrix} \dfrac{\partial N_i}{\partial r} & 0 & \dfrac{\partial N_i}{\partial z} & 0 & \dfrac{N_i}{r} \\ 0 & \dfrac{\partial N_i}{\partial r} & 0 & \dfrac{\partial N_i}{\partial z} & 0 \end{bmatrix}^T \tag{11.20}$$

for axisymmetric problems.

Note that all of the yield criteria described in Chapter 7 are included in program MIXDYN.

11.5 Program MIXDYN

11.5.1 Introduction

The computer program 'MIXDYN' is based on the Implicit–Explicit time integration scheme of Hughes and Liu[4] for two-dimensional plane stress/strain and axisymmetric nonlinear dynamic transient problems. Some of the subroutines are the same as in DYNPAK. The profile solvers DECOMP and REDBAK and a few other subroutines used in this program are based on those given in Reference (9). (These subroutines are rewritten using new variables names). Some new subroutines have also been included in the program. The program considers geometric or elasto-plastic material nonlinearity. A total Lagrangian formulation using four-, eight- and nine-noded quadrilateral isoparametric elements is adopted to model the geometric nonlinear behaviour. The program has several options; it can be used for small or large deformation elastic and small deformation elasto-plastic transient dynamic analysis and the analysis may be carried out using an explicit, implicit or combined implicit–explicit algorithm. Furthermore, four types of elasto-plastic material models can be considered: (i) Tresca, (ii) Von Mises, (iii) Drucker–Prager and (iv) Mohr–Coulomb.

The flow diagram for MIXDYN is shown in Fig. 11.3. The program is written in modular form and the input and output data representation is identical to that given for DYNPAK.

The subroutines which have not appeared elsewhere in the book are now described.

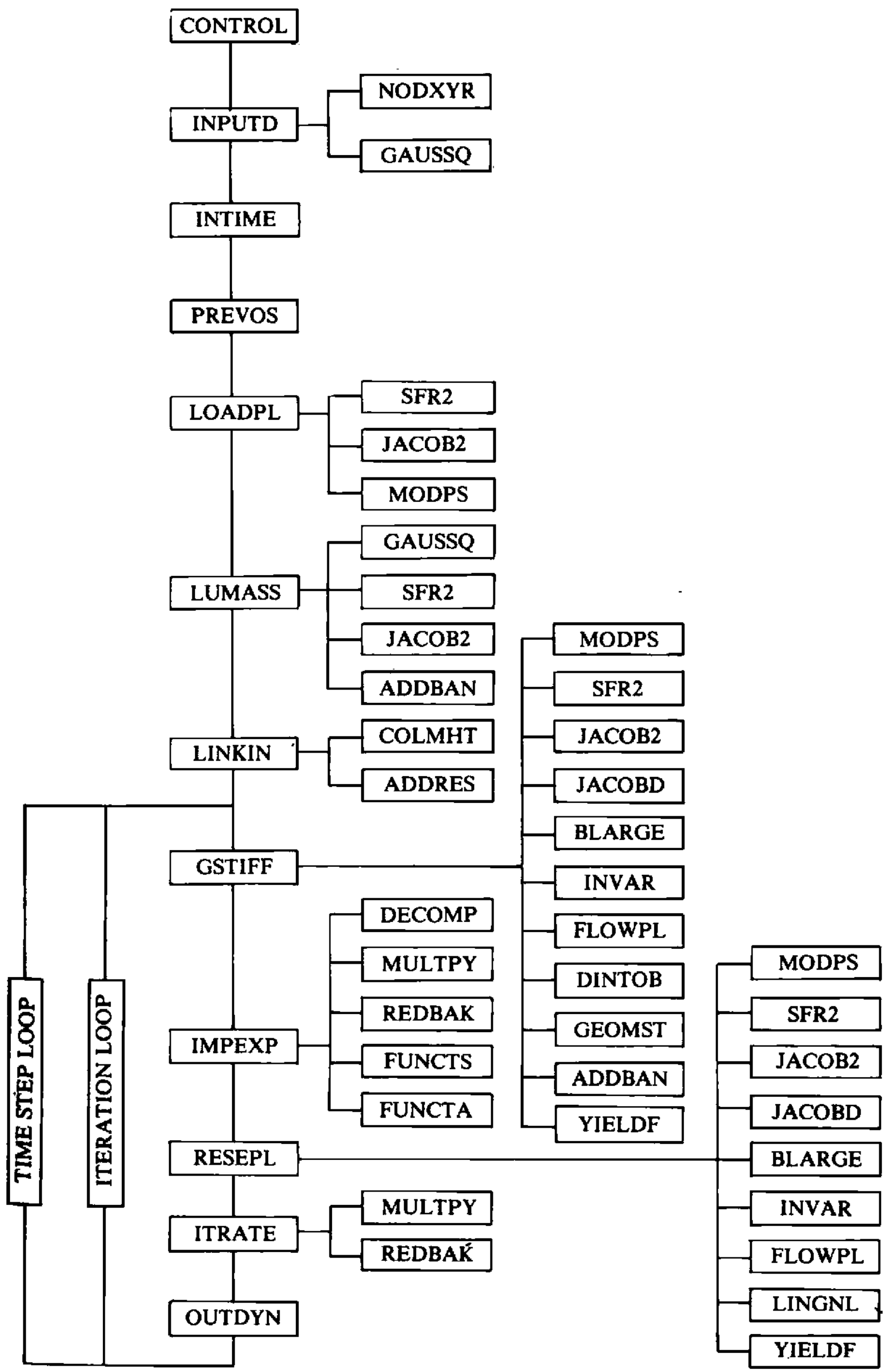


Fig. 11.3 Overall structure of program MIXDYN.

11.5.2 Master routine MIXDYN

The master routine organises the calling of the main routines as outlined in the flow diagram (Fig. 11.3). In subroutine CONTOL control parameters are read and a check is made on the maximum control dimensions. Note that the values used for checking in CONTOL should agree with the maximum dimensions in the master routine. Subroutine INPUTD, INTIME and PREVOS read the mesh data, time integration data and data for the previous state of the structure. Subroutine LINKIN links the rest of the program with the profile solver, i.e., it generates all information required for the profile solver. Subroutines LUMASS and LOADPL generate the lumped mass and applied force vectors respectively. GSTIFF calculates the global stiffness matrix in compacted form. In the time step do loop IMPEXP performs the direct time integration using either of the (i) Implicit, (ii) Explicit or (iii) combined Implicit–Explicit schemes. RESEPL calculates the equivalent nodal forces using elasto-plastic material behaviour. The maximum dimension of the program have been set to a maximum of 50 elements, 200 nodes, 10 sets of material properties, 6000 coefficients in the mass and stiffness matrices and 400 acceleration ordinates. For larger problems the dimensions must therefore be changed.

```
      PROGRAM MIXDYN (INPUT ,TAPE5=INPUT ,TAPE4,TAPE10,TAPE12,TAPE3,
     .                OUTPUT,TAPE6=OUTPUT,TAPE7,TAPE11,TAPE13)
C***********************************************************************
C
C     TIME INTEGRATION IMPLICIT-EXPLICIT ALGORITHM
C
C***********************************************************************
      DIMENSION COORD(200,2) ,STIFF(6000) ,DISPI(400) ,POSGP(  4) ,
     .          IFPRE(2,200) ,STIFS(6000) ,VELOI(400) ,WEIGP(  4) ,
     .          LNODS(50, 9) ,STIFI(6000) ,ACCEI(400) ,NPRQD( 10) ,
     .          RLOAD(50,18) ,XMASS(6000) ,DISPL(400) ,NGRQS( 10) ,
     .          PROPS(10,13) ,DAMPI(6000) ,VELOL(400) ,INTGR( 50) ,
     .          LEQNS(18,50) ,DAMPG(6000) ,ACCEL(400) ,MATNO( 50) ,
     .          STRIN(4,450) ,YMASS( 400) ,ACCEK(400) ,MAXAI(400) ,
     .          STRAG(4,450) ,FORCE( 400) ,ACCEJ(400) ,MAXAJ(400) ,
     .          STRSG(4,450) ,RESID( 400) ,DISPT(400) ,ACCEH(400) ,
     .          EPSTN(  450) ,TEMPE( 400) ,DISPQ(400) ,ACCEV(400) ,
     .          NITER( 2000) ,MHIGH( 400) ,EFFST(450) ,VELOT(400)
C
                COMMON  STIFF ,XMASS ,DAMPG ,STIFI ,STIFS ,DAMPI
C
C
      CALL      CONTOL (NDOFN ,NELEM ,NMATS ,NPOIN )
C
      CALL      INPUTD (COORD ,IFPRE ,LNODS ,MATNO ,NCONM ,NCRIT ,
     .                  NDIME ,NDOFN ,NELEM ,NGAUM ,NGAUS ,NLAPS ,
     .                  NMATS ,NNODE ,NPOIN ,NPREV ,NSTRE ,NTYPE ,
     .                  POSGP ,PROPS ,WEIGP )
C
      CALL      INTIME (AALFA ,ACCEH ,ACCEV ,AFACT ,AZERO ,BEETA ,
     .                  BZERO ,DELTA ,DTIME ,DTEND ,GAAMA ,IFIXD ,
     .                  IFUNC ,INTGR ,KSTEP ,MITER ,NDOFN ,NELEM ,
     .                  NGRQS ,NOUTD ,NOUTP ,NPOIN ,NPRQD ,NREQD ,
     .                  NREQS ,NSTEP ,OMEGA ,DISPI ,TOLER ,VELOI ,
     .                  IPRED )
```

```
C
      CALL        PREVOS (FORCE ,NDOFN ,NELEM ,NGAUS ,NPOIN ,NPREV ,
     .                   STRIN )
C
      CALL        LOADPL (COORD ,FORCE ,LNODS ,MATNO ,NDIME ,NDOFN ,
     .                   NELEM ,NGAUS ,NMATS ,NNODE ,NPOIN ,NSTRE ,
     .                   NTYPE ,POSGP ,PROPS ,RLOAD ,STRIN ,TEMPE ,
     .                   WEIGP )
C
      CALL        LUMASS (COORD ,INTGR ,LNODS ,MATNO ,NCONM ,NDIME ,
     .                   NDOFN ,NELEM ,NGAUM ,NMATS ,NNODE ,NPOIN ,
     .                   NTYPE ,PROPS ,YMASS )
C
      CALL        LINKIN (FORCE ,IFPRE ,INTGR ,LEQNS ,LNODS ,MAXAI ,
     .                   MAXAJ ,MHIGH ,NDOFN ,NELEM ,NEQNS ,NNODE ,
     .                   NPOIN ,NWKTL ,NWMTL ,XMASS ,YMASS)
C
      DO 510 ISTEP=1,NSTEP
C
      DO 500 IITER=1,MITER
C
      CALL        GSTIFF (COORD ,EPSTN ,INTGR ,ISTEP ,KSTEP ,LEQNS ,
     .                   LNODS ,MATNO ,MAXAI ,MAXAJ ,NCRIT ,NDIME ,
     .                   NDOFN ,NELEM ,NGAUS ,NLAPS ,NMATS ,NNODE ,
     .                   NPOIN ,NSTRE ,NTYPE ,NWMTL ,NWKTL ,POSGP ,
     .                   PROPS ,STIFF ,STIFI ,STRSG ,DISPT ,WEIGP )
C
      CALL        IMPEXP (AALFA ,ACCEH ,ACCEI ,ACCEJ ,ACCEK ,ACCEL ,
     .                   ACCEV ,AFACT ,AZERO ,BEETA ,BZERO ,CONSD ,
     .                   CONSF ,DAMPI ,DAMPG ,DELTA ,DISPI ,DISPL ,
     .                   DISPT ,DTEND ,DTIME ,GAAMA ,IFIXD ,IFPRE ,
     .                   IFUNC ,IITER ,ISTEP ,KSTEP ,MAXAI ,MAXAJ ,
     .                   NDOFN ,NEQNS ,NPOIN ,NWKTL ,NWMTL ,OMEGA ,
     .                   FORCE ,STIFF ,STIFI ,STIFS ,VELOI ,VELOL ,
     .                   VELOT ,XMASS ,YMASS ,IPRED )
C
      CALL        RESEPL (COORD ,DISPT ,EFFST ,RLOAD ,EPSTN ,IITER ,
     .                   INTGR ,LEQNS ,LNODS ,MATNO ,NCRIT ,NDIME ,
     .                   NDOFN ,NELEM ,NGAUS ,NLAPS ,NMATS ,NNODE ,
     .                   NPOIN ,NSTRE ,NTYPE ,POSGP ,PROPS ,RESID ,
     .                   STRAG ,STRIN ,STRSG ,WEIGP ,IPRED ,ISTEP )
C
      CALL        ITRATE (ACCEI ,ACCEL ,CONSD ,CONSF ,XMASS ,DISPI ,
     .                   DISPL ,DISPT ,MAXAI ,NCHEK ,NEQNS ,NWMTL ,
     .                   RESID ,STIFS ,TOLER ,VELOI ,VELOL ,VELOT ,
     .                   IITER ,MITER )
C
  500 IF(NCHEK.EQ.1) GO TO 510
C
  510 CALL        OUTDYN (DISPQ ,DTIME ,EPSTN ,IFPRE ,IITER ,ISTEP ,
     .                   NDOFN ,NELEM ,NGAUS ,NGRQS ,NITER ,NOUTD ,
     .                   NOUTP ,NPOIN ,NPRQD ,NREQD ,NREQS ,NTYPE ,
     .                   STRSG ,DISPI )
C
      STOP
      END
```

11.5.3 Subroutine ADDBAN

This routine[9] assembles the element stiffness matrix into the global stiffness matrix in a compacted form.

```
      SUBROUTINE ADDBAN (STIFF,MAXAI,ESTIF,LEQNS,NEVAB)
C*********************************************************************
C
C *** ASSEMBLY OF TOTAL STIFFNESS VECTOR
C
C*********************************************************************
      DIMENSION STIFF(1),MAXAI(1),ESTIF(1),LEQNS(1)
C
      KOUNT=0
      DO 200 IEVAB=1,NEVAB
      IEQNS=LEQNS(IEVAB)
      IF(IEQNS) 200,200,100
  100 IMAXA=MAXAI(IEQNS)
      KEVAB=IEVAB
      DO 220 JEVAB=1,NEVAB
      JEQNS=LEQNS(JEVAB)
      IF(JEQNS) 220,220,110
  110 IJEQN=IEQNS-JEQNS
      IF(IJEQN) 220,210,210
  210 ISIZE=IMAXA+IJEQN
      JSIZE=KEVAB
      IF(JEVAB.GE.IEVAB) JSIZE=JEVAB+KOUNT
      STIFF(ISIZE)=STIFF(ISIZE)+ESTIF(JSIZE)
  220 KEVAB=KEVAB+NEVAB-JEVAB
  200 KOUNT=KOUNT+NEVAB-IEVAB
      RETURN
      END
```

11.5.4 Subroutine ADDRES

This routine[9] addresses the diagonal elements of the global matrix using the column heights.

```
      SUBROUTINE ADDRES(MAXAI ,MHIGH ,NEQNS ,NWKTL ,MKOUN )
C*********************************************************************
C
C *** EVALUATES ADRESSES OF DIAGONAL ELEMENTS
C
C*********************************************************************
      DIMENSION MAXAI(1) ,MHIGH(1)
      NEQNN=NEQNS+1
      DO 20 IEQNN=1,NEQNN
   20 MAXAI(1)=1
      MAXAI(2)=2
      MKOUN=0
      IF(NEQNS.EQ.1) GO TO 30
      DO 10 IEQNS=2,NEQNS
      IF(MHIGH(IEQNS).GT.MKOUN) MKOUN=MHIGH(IEQNS)
   10 MAXAI(IEQNS+1)=MAXAI(IEQNS)+MHIGH(IEQNS)+1
   30 MKOUN=MKOUN+1
      NWKTL=MAXAI(NEQNS+1)-MAXAI(1)
      RETURN
      END
```

11.5.5 Subroutine COLMHT

This routine[9] calculates the vertical column heights above the diagonal of the global matrix using equation numbers and the total number of degrees of freedom of an element (NEVAB).

```
      SUBROUTINE COLMHT (MHIGH ,NEVAB ,LEQNS )
C*********************************************************************
C
C*** EVALUATES THE COLUMN HEIGHT OF STIFFNESS MATRIX
C
C*********************************************************************
      DIMENSION LEQNS(1) ,MHIGH(1)
      MAXAM=100000
      DO 100 IEVAB=1,NEVAB
      IF(LEQNS(IEVAB)) 110,100,110
  110 IF(LEQNS(IEVAB)-MAXAM) 120,100,100
  120 MAXAM=LEQNS(IEVAB)
  100 CONTINUE
      DO 200 IEVAB=1,NEVAB
      IEQNS=LEQNS(IEVAB)
      IF(IEQNS.EQ.0) GO TO 200
      JHIGH=IEQNS-MAXAM
      IF(JHIGH.GT.MHIGH(IEQNS)) MHIGH(IEQNS)=JHIGH
  200 CONTINUE
      RETURN
      END
```

11.5.6 Subroutine DECOMP

This routine[9] factorises a matrix into lower, diagonal and upper matrices (LDL^T)

```
      SUBROUTINE DECOMP (STIFF ,MAXAI ,NEQNS ,ISHOT )
C*********************************************************************
C
C *** FACTORISES (L)*(D)*(L) TRANSPOSE OF STIFFNESS MATRIX
C
C*********************************************************************
      DIMENSION STIFF(1) ,MAXAI(1)
C
      IF(NEQNS.EQ.1) RETURN
      DO 200 IEQNS=1,NEQNS
      IMAXA=MAXAI(IEQNS)
      LOWER=IMAXA+1
      KUPER=MAXAI(IEQNS+1)-1
      KHIGH=KUPER-LOWER
      IF(KHIGH) 304,240,210
  210 KSIZE=IEQNS-KHIGH
      ICOUN=0
      JUPER=KUPER
      DO 260 JHIGH=1,KHIGH
      ICOUN=ICOUN+1
      JUPER=JUPER-1
      KMAXA=MAXAI(KSIZE)
      NDIAG=MAXAI(KSIZE+1)-KMAXA-1
      IF(NDIAG) 260,260,270
```

```
  270 NCOLM=MINO(ICOUN,NDIAG)
      COUNT=0.
      DO 280 ICOLM=1,NCOLM
  280 COUNT=COUNT+STIFF(KMAXA+ICOLM)*STIFF(JUPER+ICOLM)
      STIFF(JUPER)=STIFF(JUPER)-COUNT
  260 KSIZE=KSIZE+1
  240 KSIZE=IEQNS
      BSUMM=0.
      DO 300 ICOLM=LOWER,KUPER
      KSIZE=KSIZE-1
      JMAXA=MAXAI(KSIZE)
      RATIO=STIFF(ICOLM)/STIFF(JMAXA)
      BSUMM=BSUMM+RATIO*STIFF(ICOLM)
  300 STIFF(ICOLM)=RATIO
      STIFF(IMAXA)=STIFF(IMAXA)-BSUMM
  304 IF(STIFF(IMAXA)) 310,310,200
  310 IF(ISHOT.EQ.0) GO TO 320
      IF(STIFF(IMAXA).EQ.0) STIFF(IMAXA)=-1.E-16
      GO TO 200
  320 WRITE(6,2000) IEQNS,STIFF(IMAXA)
      STOP
  200 CONTINUE
      RETURN
 2000 FORMAT(//48H STOP - STIFFNESS MATRIX NOT POSITIVE DEFINITE  ,//
     .32H NONPOSITIVE PIVOT FOR EQUATION ,I4,//10H PIVOT =  ,E20.12 )
      END
```

11.5.7 Subroutine DINTOB

This routine multiplies the modulus matrix $\boldsymbol{D}$ with the strain matrix $\boldsymbol{B}$.

```
      SUBROUTINE DINTOB (BMATX ,DBMAT ,DMATX ,NEVAB ,NSTRE )
C*********************************************************************
C
C***  CALCULATE D INTO B
C
C*********************************************************************
      DIMENSION DBMAT(4,18),DMATX(4,4),BMATX(4,18)
      DO 10 ISTRE=1,NSTRE
      DO 10 IEVAB=1,NEVAB
      DBMAT(ISTRE,IEVAB)=0.0
      DO 10 JSTRE=1,NSTRE
      DBMAT(ISTRE,IEVAB)=DBMAT(ISTRE,IEVAB)+
     .DMATX(ISTRE,JSTRE)*BMATX(JSTRE,IEVAB)
   10 CONTINUE
      RETURN
      END
```

11.5.8 Subroutine GEOMST

This routine adds the initial stress matrix to the stiffness matrix.

```
      SUBROUTINE GEOMST (CARTD ,DVOLU ,ESTIF ,KGAUS ,NDOFN ,NNODE ,
     .                   STRSG ,SHAPE ,NTYPE ,GPCOD ,KGASP )
C*********************************************************************
C
C       ADD INITIAL STRESS STIFFNESS MATRIX TO STIFFNESS MATRIX
C
C*********************************************************************
      DIMENSION STRES(4) ,CARTD(2,9) ,ESTIF(171) ,STRSG(4,1) ,
     .          SHAPE(1) ,GPCOD(2,9)
      NEVAB=NNODE*NDOFN
      DO 300 ISTR1=1,4
```

```
  300 STRES(ISTR1)=STRSG(ISTR1,KGAUS)
      IEVAB=1
      KOUNT=NEVAB
      DO 200 INODE=1,NNODE
      DO 100 JNODE=INODE,NNODE
      DGASH=STRES(1)*CARTD(1,INODE)*CARTD(1,JNODE)+
     .STRES(3)*(CARTD(1,INODE)*CARTD(2,JNODE)+
     .CARTD(2,INODE)*CARTD(1,JNODE))+
     .STRES(2)*CARTD(2,INODE)*CARTD(2,JNODE)
      DGASY=DGASH*DVOLU
      DGASX=DGASY
      IF(NTYPE.NE.3) GO TO 400
      PRODT=SHAPE(INODE)/(GPCOD(1,KGASP)**2)
      DGASX=DGASY+STRES(4)*PRODT*SHAPE(JNODE)*DVOLU
  400 ESTIF(IEVAB)=ESTIF(IEVAB)+DGASX
      JEVAB=IEVAB+KOUNT
      ESTIF(JEVAB)=ESTIF(JEVAB)+DGASY
      IEVAB=IEVAB+2
  100 CONTINUE
      KOUNT=KOUNT-2
      IEVAB=JEVAB+1
  200 CONTINUE
      RETURN
      END
```

11.5.9 Subroutine GSTIFF

This routine generates the compacted geometrically nonlinear stiffness matrix for two-dimensional plane stress/strain and axisymmetric problems from the element stiffness matrices.

```
      SUBROUTINE GSTIFF (COORD ,EPSTN ,INTGR ,ISTEP ,KSTEP ,LEQNS ,
     .                   LNODS ,MATNO ,MAXAI ,MAXAJ ,NCRIT ,NDIME ,
     .                   NDOFN ,NELEM ,NGAUS ,NLAPS ,NMATS ,NNODE ,
     .                   NPOIN ,NSTRE ,NTYPE ,NWMTL ,NWKTL ,POSGP ,
     .                   PROPS ,STIFF ,STIFI ,STRSG ,TDISP ,WEIGP )
C***********************************************************************
C
C       EVALUATES GEOMETRICALLY NONLINEAR STIFFNESS MATRIX
C       FOR 2-D PLANE STRESS/STRAIN 2-D ELEMENT
C
C***********************************************************************
      DIMENSION COORD(NPOIN,1) ,DMATX(4, 4) ,ELCOD(2,9) ,AVECT(4) ,
     .          LNODS(NELEM,1) ,BMATX(4,18) ,CARTD(2,9) ,DVECT(4) ,
     .          PROPS(NMATS,1) ,DBMAT(4,18) ,GPCOD(2,9) ,DEVIA(4) ,
     .          LEQNS(  18, 1) ,STRSG(4, 1) ,DLCOD(2,9) ,STRES(4) ,
     .          ESTIF(    171) ,DJACM(2, 2) ,DERIV(2,9) ,SHAPE(9)
C
      DIMENSION MAXAI(1) ,INTGR(1) ,STIFF(1) ,POSGP(1) ,EPSTN(1) ,
     .          MAXAJ(1) ,TDISP(1) ,STIFI(1) ,WEIGP(1) ,MATNO(1)
C
C
      IF(ISTEP.EQ.1) GO TO 200
      KOUNT=(ISTEP/KSTEP)*KSTEP
      IF(KOUNT.NE.ISTEP)RETURN
  200 CONTINUE
      TWOPI=6.283185307179586
      KGAUS=0
C
C*** LOOP OVER EACH ELEMENT
C
      NSTR1=4
      NEVAB=NDOFN*NNODE
      DO 500 IWKTL=1,NWKTL
  500 STIFF(IWKTL) ,STIFI(IWKTL)=0.0
```

```
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      IPOSN=0
      DO 10 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      DO 10 IDIME=1,NDIME
      IPOSN=IPOSN+1
      NPOSN=LEQNS(IPOSN,IELEM)
      IF(NPOSN.EQ.0) DISPT=0.
      IF(NPOSN.NE.0) DISPT=TDISP(NPOSN)
      DLCOD(IDIME,INODE)=COORD(LNODE,IDIME)+DISPT
   10 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
      YOUNG=PROPS(LPROP, 1)
      POISS=PROPS(LPROP, 2)
      THICK=PROPS(LPROP, 3)
      HARDS=PROPS(LPROP, 7)
      FRICT=PROPS(LPROP, 8)
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX 171=NEVAB*(NEVAB+1)/2
C
      DO 20 ISIZE=1,171
   20 ESTIF(ISIZE)=0.0
      KGASP=0
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      DO 50 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      DO 50 JGAUS=1,NGAUS
      ETASP=POSGP(JGAUS)
      KGASP=KGASP+1
      KGAUS=KGAUS+1
      CALL MODPS  (DMATX,LPROP,NMATS,NSTRE,NTYPE,PROPS)
      CALL SFR2   (DERIV,NNODE,SHAPE,EXISP,ETASP)
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,
     .             IELEM,KGASP,NNODE,SHAPE)
      CALL JACOBD (CARTD,DLCOD,DJACM,NDIME,NLAPS,NNODE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(NTYPE.EQ.1) DVOLU=DVOLU*THICK
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL BLARGE (BMATX,CARTD,DJACM,DLCOD,GPCOD,
     .             KGASP,NLAPS,NNODE,NTYPE,SHAPE)
      IF(NLAPS.EQ.2.OR.NLAPS.EQ.0) GO TO 80
      IF(ISTEP.EQ.1) GO TO 80
      IF(EPSTN(KGAUS).EQ.0.0) GO TO 80
      DO 90 ISTR1=1,NSTR1
   90 STRES(ISTR1)=STRSG(ISTR1,KGAUS)
      CALL INVAR  (DEVIA,LPROP,NCRIT,NMATS,PROPS,SINT3,STEFF,
     .             STRES,THETA,VARJ2,YIELD)
      CALL YIELDF (AVECT,DEVIA,FRICT,NCRIT,SINT3,STEFF,
     .             THETA,VARJ2)
      CALL FLOWPL (AVECT,ABETA,DVECT,HARDS,NTYPE,POISS,YOUNG)
      DO 100 ISTRE=1,NSTRE
      DO 100 JSTRE=1,NSTRE
  100 DMATX(ISTRE,JSTRE)=DMATX(ISTRE,JSTRE)-ABETA*DVECT(ISTRE)*
     . DVECT(JSTRE)
   80 CONTINUE
      CALL DINTOB (BMATX,DBMAT,DMATX,NEVAB,NSTRE)
```

```
C
C ***EVALUATE GEOMETRIC STIFFNESS TERMS
C
      IF(NLAPS.LT.2) GO TO 85
      CALL GEOMST (CARTD,DVOLU,ESTIF,KGAUS,NDOFN,NNODE,
     .             STRSG,SHAPE,NTYPE,GPCOD,KGASP)
C
C*** CALCULATE THE ELEMENT STIFFNESSES
C
   85 KOUNT=0
      DO 30 IEVAB=1,NEVAB
      DO 30 JEVAB=IEVAB,NEVAB
      KOUNT=KOUNT+1
      DO 30 ISTRE=1,NSTRE
   30 ESTIF(KOUNT)=ESTIF(KOUNT)+BMATX(ISTRE,IEVAB)*
     .             DBMAT(ISTRE,JEVAB)*DVOLU
   50 CONTINUE
C
C *** GENERATES GLOBAL STIFFNSS MATRIX IN COMPACTED COLUMN FORM
C
      IF(INTGR(IELEM).EQ.2) GO TO 210
      CALL ADDBAN (STIFI,MAXAI,ESTIF,LEQNS(1,IELEM),NEVAB)
  210 CALL ADDBAN (STIFF,MAXAJ,ESTIF,LEQNS(1,IELEM),NEVAB)
   70 CONTINUE
C     WRITE(6,900) (STIFI(I),I=1,NWMTL)
  900 FORMAT(10E12.4)
      RETURN
      END
```

11.5.10 Subroutine IMPEXP

This routine generates the partial effective load vector for direct time integration.

```
      SUBROUTINE IMPEXP (AALFA ,ACCEH ,ACCEI ,ACCEJ ,ACCEK ,ACCEL ,
     .                   ACCEV ,AFACT ,AZERO ,BEETA ,BZERO ,CONSD ,
     .                   CONSF ,DAMPI ,DAMPG ,DELTA ,DISPI ,DISPL ,
     .                   DISPT ,DTEND ,DTIME ,GAAMA ,IFIXD ,IFPRE ,
     .                   IFUNC ,IITER ,ISTEP ,KSTEP ,MAXAI ,MAXAJ ,
     .                   NDOFN ,NSIZE ,NPOIN ,NWKTL ,NWMTL ,OMEGA ,
     .                   RLOAD ,STIFF ,STIFI ,STIFS ,VELOI ,VELOL ,
     .                   VELOT ,XMASS ,YMASS ,IPRED )
C*****************************************************************
C
C *** GENERATES PARTIAL EFFECTIVE LOAD VECTOR
C
C*****************************************************************
      DIMENSION STIFF(1) ,DISPI(1) ,ACCEH(1) ,DISPL(1) ,IFPRE(2,1) ,
     .          XMASS(1) ,VELOI(1) ,ACCEV(1) ,VELOL(1) ,ACCEK(  1) ,
     .          RLOAD(1) ,ACCEI(1) ,MAXAI(1) ,ACCEL(1) ,DAMPG(  1) ,
     .          ACCEJ(1) ,MAXAJ(1) ,YMASS(1) ,STIFI(1) ,DISPT(  1) ,
     .          STIFS(1) ,DAMPI(1) ,VELOT(1)
C
C
C
      IF(ISTEP.GT.1.OR.IITER.GT.1) GO TO 1000
      CONSA=DTIME*DTIME*(0.5-DELTA)
      CONSB=DTIME*(1.-GAAMA)
      CONSC=DTIME*DTIME*DELTA
      CONSD=DTIME*GAAMA
      CONSF=1./CONSC
      CONSG=BEETA*GAAMA*DTIME
      CONSH=AALFA*GAAMA*DTIME
      CONSE=1.+CONSH
```

```
      ISHOT=0
      DO 550 IPOIN=1,NPOIN
      DO 550 IDOFN=1,NDOFN
      ISIZE=IFPRE(IDOFN,IPOIN)
      IF(ISIZE.EQ.0) GO TO 550
      ACCEI(ISIZE)=1.0
      ACCEL(ISIZE)=0.0
      IF(IDOFN.EQ.1) GO TO 550
      ACCEI(ISIZE)=0.0
      ACCEL(ISIZE)=1.0
  550 CONTINUE
      DO 590 ISIZE=1,NSIZE
      IMAXA=MAXAI(ISIZE)
  590 XMASS(IMAXA)=XMASS(IMAXA)+YMASS(ISIZE)
C
C *** CALCULATES VECTORS FOR HORIZONTAL AND VERTICAL EXCITATION
C
      CALL MULTPY (ACCEK,XMASS,ACCEL,MAXAI,NSIZE,NWMTL)
      CALL MULTPY (ACCEJ,XMASS,ACCEI,MAXAI,NSIZE,NWMTL)
      CALL MULTPY (DISPL,STIFF,DISPI,MAXAJ,NSIZE,NWKTL)
C
C *** CALCULATES DAMPING MATRIX (AALFA*M+BEETA*K)
C
      DO 500 ISIZE=1,NSIZE
      IMAXA=MAXAI(ISIZE)
      KMAXA=MAXAI(ISIZE+1)-1
      JMAXA=MAXAJ(ISIZE)
      DO 500 LMAXA=IMAXA,KMAXA
      DAMPI(JMAXA)=AALFA*XMASS(LMAXA)
  500 JMAXA=JMAXA+1
      DO 560 IWKTL=1,NWKTL
  560 DAMPI(IWKTL)=DAMPI(IWKTL)+BEETA*STIFF(IWKTL)
C
C *** CALCULATES INITIAL ACCELERATION
C
      CALL MULTPY (VELOL,DAMPI,VELOI,MAXAJ,NSIZE,NWKTL)
      DO 600 IWMTL=1,NWMTL
  600 DAMPG(IWMTL)=XMASS(IWMTL)
      DO 510 ISIZE = 1,NSIZE
  510 ACCEI(ISIZE)=RLOAD(ISIZE)-DISPL(ISIZE)-VELOL(ISIZE)
      CALL DECOMP (DAMPG,MAXAI,NSIZE,ISHOT)
      CALL REDBAK (DAMPG,ACCEI,MAXAI,NSIZE)
      WRITE (6,900)
      WRITE (6,910) (ACCEI(ISIZE),ISIZE=1,NSIZE)
  900 FORMAT(/'  INITIAL ACCELERATION '/)
  910 FORMAT(1X,10E12.5)
 1000 CONTINUE
      IF(IITER.GT.1) GO TO 650
C
C *** CALCULATES PREDICTED DISPLACEMENT AND VELOCITY VECTOR
C
      DO 540 ISIZE=1,NSIZE
      IF(IPRED.EQ.1) GO TO 210
      DISPT(ISIZE)=DISPI(ISIZE)
      VELOT(ISIZE)=VELOI(ISIZE)
  210 DISPI(ISIZE)=DISPI(ISIZE)+DTIME*VELOI(ISIZE)+CONSA*ACCEI(ISIZE)
      VELOI(ISIZE)=VELOI(ISIZE)+CONSB*ACCEI(ISIZE)
      IF(IPRED.EQ.2) GO TO 220
      DISPT(ISIZE)=DISPI(ISIZE)
      VELOT(ISIZE)=VELOI(ISIZE)
  220 ACCEI(ISIZE)=CONSF*(DISPT(ISIZE)-DISPI(ISIZE))
  540 CONTINUE
C
C*** CALCULATES LOAD VECTORS
```

```
C
      FACTS =FUNCTS (AZERO,BZERO,DTEND,DTIME,IFUNC,ISTEP,OMEGA)
      FACTH =FUNCTA (ACCEH,AFACT,DTEND,DTIME,IFUNC,ISTEP)
      FACTV =FUNCTA (ACCEV,AFACT,DTEND,DTIME,IFUNC,ISTEP)
C     WRITE(6,910) FACTS,FACTH,FACTV
  650 CONTINUE
      IF(ISTEP.EQ.1) GO TO 640
C
C *** CALCULATES DAMPING AND K-STAR MATRICES
C
      DO 530 ISIZE=1,NSIZE
      IMAXA=MAXAI(ISIZE)
      KMAXA=MAXAI(ISIZE+1)-1
      JMAXA=MAXAJ(ISIZE)
      DO 530 LMAXA=IMAXA,KMAXA
      DAMPI(JMAXA)=AALFA*XMASS(LMAXA)
  530 JMAXA=JMAXA+1
      DO 580 IWKTL=1,NWKTL
  580 DAMPI(IWKTL)=DAMPI(IWKTL)+BEETA*STIFF(IWKTL)
      CALL MULTPY (VELOL ,DAMPI ,VELOT ,MAXAJ ,NSIZE ,NWKTL )
      KOUNT=(ISTEP/KSTEP)*KSTEP
      IF(KOUNT.NE.ISTEP) GO TO 660
  640 DO 610 IWMTL=1,NWMTL
  610 DAMPG(IWMTL)=CONSE*XMASS(IWMTL)
      DO 620 ISIZE=1,NSIZE
      IMAXA=MAXAI(ISIZE)
  620 DAMPG(IMAXA)=DAMPG(IMAXA)-CONSH*YMASS(ISIZE)
      DO 630 IWMTL=1,NWMTL
      DAMPG(IWMTL)=DAMPG(IWMTL)+CONSG*STIFI(IWMTL)
  630 STIFS(IWMTL)=STIFI(IWMTL)+DAMPG(IWMTL)*CONSF
C     WRITE(6,900) (STIFS(I),I=1,NWMTL)
      CALL DECOMP (STIFS ,MAXAI ,NSIZE ,ISHOT )
C
C *** CALCULATES PARTIAL EFFECTIVE LOAD VECTOR
C
  660 DO 520 ISIZE=1,NSIZE
      IF(IFUNC.NE.0) GO TO 570
      IF(IFIXD.EQ.2) DISPL(ISIZE)=-VELOL(ISIZE)-FACTH*ACCEJ(ISIZE)
     .                            +RLOAD(ISIZE)
      IF(IFIXD.EQ.1) DISPL(ISIZE)=-VELOL(ISIZE)-FACTV*ACCEK(ISIZE)
     .                            +RLOAD(ISIZE)
      IF(IFIXD.EQ.0) DISPL(ISIZE)=-VELOL(ISIZE)-FACTH*ACCEJ(ISIZE)
     .                            +RLOAD(ISIZE)-FACTV*ACCEK(ISIZE)
      IF(IFUNC.EQ.0) GO TO 520
  570 DISPL(ISIZE)=-VELOL(ISIZE)+RLOAD(ISIZE)*FACTS
  520 CONTINUE
      RETURN
      END
```

11.5.11 Subroutine ITRATE

This routine generates the total effective load vector and solves for the incremental displacements. It then checks for convergence.

```
      SUBROUTINE ITRATE (ACCEI ,ACCEL ,CONSD ,CONSF ,XMASS ,DISPI ,
     .                   DISPL ,DISPT ,MAXAI ,NCHEK ,NSIZE ,NWMTL ,
     .                   RESID ,STIFS ,TOLER ,VELOI ,VELOL ,VELOT ,
     .                   IITER ,MITER )
C*********************************************************************
C
C *** CALCULATES INCREMENT IN DISPLACEMENT AND APPLIES CONVERGENCE
C
C*********************************************************************
      DIMENSION DISPI(1) ,VELOI(1) ,ACCEI(1) ,RESID(1) ,MAXAI(1) ,
```

```
ITER 11           .    DISPL(1) ,VELOL(1) ,ACCEL(1) ,STIFS(1) ,DISPT(1) ,
ITER 12           .    XMASS(1) ,VELOT(1)
ITER 13 C
ITER 14 C
ITER 15       NCHEK=0
ITER 16       CALL MULTPY (ACCEL ,XMASS ,ACCEI ,MAXAI ,NSIZE ,NWMTL )
ITER 17 C
ITER 18 C *** CALCULATES TOTAL EFFECTIVE LOAD VECTOR
ITER 19 C
ITER 20       DO 660 ISIZE=1,NSIZE
ITER 21   660 ACCEL(ISIZE)=DISPL(ISIZE)-ACCEL(ISIZE)-RESID(ISIZE)
ITER 22 C
ITER 23 C *** CALCULATES DELTA DISPLACEMENT
ITER 24 C
ITER 25   210 CALL REDBAK (STIFS,ACCEL,MAXAI,NSIZE)
ITER 26 C
ITER 27 C *** APPLIES CONVERGENCE
ITER 28 C
ITER 29       SUMPP=0.
ITER 30       SUMPQ=0.
ITER 31       DO 670 ISIZE=1,NSIZE
ITER 32       DISPP=ACCEL(ISIZE)
ITER 33       DISPQ=DISPT(ISIZE)+DISPP
ITER 34       DISPT(ISIZE)=DISPQ
ITER 35       SUMPP=SUMPP+DISPP*DISPP
ITER 36       SUMPQ=SUMPQ+DISPQ*DISPQ
ITER 37   670 CONTINUE
ITER 38       DO 530 ISIZE=1,NSIZE
ITER 39       ACCEI(ISIZE)=CONSF*(DISPT(ISIZE)-DISPI(ISIZE))
ITER 40   530 VELOT(ISIZE)=VELOI(ISIZE)+CONSD*ACCEI(ISIZE)
ITER 41   220 SUMPP=SQRT(SUMPP/SUMPQ)
ITER 42       IF(SUMPP.GT.TOLER) GO TO 550
ITER 43       NCHEK=1
ITER 44       GO TO 240
ITER 45   550 IF(IITER.LT.MITER) GO TO 230
ITER 46   240 DO 540 ISIZE=1,NSIZE
ITER 47       VELOI(ISIZE)=VELOT(ISIZE)
ITER 48   540 DISPI(ISIZE)=DISPT(ISIZE)
ITER 49   230 CONTINUE
ITER 50       RETURN
ITER 51       END
```

ITER 20–21	Calculates total effective load vector.
ITER 25	Solves for incremental displacements.
ITER 28–37	Calculates norm of displacement increments.
ITER 38–40	Calculates new and total displacement, velocities and accelerations.
ITER 41–42	Applies convergence check.
ITER 46–49	Stores the final velocities and displacements in vectors VELOI and DISPI respectively.

11.5.12 Subroutine LINKIN

This routine calculates the equation number from the array IFPRE which stores the information about the restrained degrees of freedom.

```
      SUBROUTINE LINKIN    (FORCE ,IFPRE ,INTGR ,LEQNS ,LNODS ,MAXAI ,
     .                      MAXAJ ,MHIGH ,NDOFN ,NELEM ,NEQNS ,NNODE ,
     .                      NPOIN ,NWKTL ,NWMTL ,XMASS ,YMASS)
C*********************************************************************
C
C ***   LINKS WITH PROFILE SOLVER
C
C*********************************************************************
      DIMENSION LNODS(NELEM,1) ,XMASS(1) ,MAXAI(1) ,INTGR(1) ,
     .          IFPRE(NDOFN,1) ,YMASS(1) ,MAXAJ(1) ,MHIGH(1) ,
     .          LEQNS(   18,1) ,FORCE(1) ,EMASS(171)
C
      IMASS=1
      REWIND 3
      NEVAB=NNODE*NDOFN
C
C***  NUMBER OF UNKNOWNS
C
      NEQNS=0
      DO 100 IPOIN=1,NPOIN
      DO 150 IDOFN=1,NDOFN
      IF(IFPRE(IDOFN,IPOIN)) 110,120,110
120   NEQNS=NEQNS+1
      IFPRE(IDOFN,IPOIN)=NEQNS
      GO TO 150
110   IFPRE(IDOFN,IPOIN)=0
 150  CONTINUE
C     WRITE(6,7) IPOIN,(IFPRE(IDOFN,IPOIN),IDOFN=1,NDOFN)
100   CONTINUE
      MEQNS=1+NEQNS
C
C***  CONNECTIVITY ARRAY LEQNS
C
      DO 70 IELEM=1,NELEM
      DO 70 IEVAB=1,NEVAB
70    LEQNS(IEVAB,IELEM)=0
      DO 50 IELEM=1,NELEM
      IEVAB=1
      DO 80 INODE=1,NNODE
      IDENT=LNODS(IELEM,INODE)
      DO 80 IDOFN=1,NDOFN
      LEQNS(IEVAB,IELEM)=IFPRE(IDOFN,IDENT)
80    IEVAB=IEVAB+1
C     WRITE(6,6) IELEM,(LEQNS(IEVAB,IELEM),IEVAB=1,NEVAB)
50    CONTINUE
6     FORMAT(I10,24I3)
7     FORMAT(4I10)
8     FORMAT(8E12.4)
C
C***  LOOP OVER ALL ELEMENTS
C
  250 DO 190 IELEM=1,NELEM
      IF(INTGR(IELEM).NE.IMASS) GO TO 190
      CALL COLMHT (MHIGH,NEVAB,LEQNS(1,IELEM))
190   CONTINUE
C
C***  ADDRESES OF DIAGONAL ELEMENTS - MAXA ARRAY
C
      CALL ADDRES(MAXAJ,MHIGH,NEQNS,NWKTL,MKOUN)
      IF(IMASS.EQ.2) GO TO 205
      DO 580 IEQNS=1,MEQNS
  580 MAXAI(IEQNS)=MAXAJ(IEQNS)
      IMASS=2
      NWMTL=NWKTL
```

```
LINK  65        GO TO 250
LINK  66    205 CONTINUE
LINK  67        WRITE(6,920) NEQNS,NWMTL,NWKTL
LINK  68        WRITE(6,930) (MAXAI(I),I=1,MEQNS)
LINK  69        WRITE(6,930) (MAXAJ(I),I=1,MEQNS)
LINK  70    930 FORMAT(5X,20I5)
LINK  71    920 FORMAT(/5X,'NEQNS=',I5,5X,'NWMTL=',I5,5X,'NWKTL=',I5/)
LINK  72        IF(NWKTL.GT.6000) GO TO 210
LINK  73        GO TO 220
LINK  74    210 WRITE(6,910)
LINK  75        STOP
LINK  76    220 CONTINUE
LINK  77    910 FORMAT (/'SET DIMENSION EXCEEDED - CHECK LINKIN '/)
LINK  78  C
LINK  79  C***  GLOBAL MASS MATRIX
LINK  80  C
LINK  81        DO 500 IELEM=1,NELEM
LINK  82        IMASS=INTGR(IELEM)
LINK  83        IF(IMASS.EQ.2) GO TO 500
LINK  84        READ (3) EMASS
LINK  85        CALL ADDBAN (XMASS,MAXAI,EMASS,LEQNS(1,IELEM),NEVAB)
LINK  86    500 CONTINUE
LINK  87  C
LINK  88  C***  GLOBAL MASS VECTOR
LINK  89  C
LINK  90        NPOSM=0
LINK  91        DO 510 IPOIN =1,NPOIN
LINK  92        DO 510 IDOFN =1,NDOFN
LINK  93        NPOSM=NPOSM+1
LINK  94        NPOSN=IFPRE(IDOFN,IPOIN)
LINK  95        IF(NPOSN.EQ.0) GO TO 510
LINK  96        YMASS(NPOSN)=YMASS(NPOSM)
LINK  97        FORCE(NPOSN)=FORCE(NPOSM)
LINK  98    510 CONTINUE
LINK  99        RETURN
LINK 100        END
```

LINK 18–29 Reassigns IFPRE vector with equation numbers. If IFPRE is not zero than IFPRE is reassigned as zero.

LINK 34–45 Evaluates the vector LEQNS on element level for assigning equation number corresponding to each node in an element.

LINK 52–55 Calculates column height above the diagonal in global matrix.

LINK 59–62 Assigns location for diagonal elements in global matrix.

LINK 80–85 IMASS = 1 calculates stiffness matrix for only implicit elements.
IMASS = 2 calculates stiffness matrix for complete mesh.

11.5.13 Subroutine MULTPY

This routine[9] evaluates the product of square matrix AMATX and an array START and stores the result in FINAL.

```
      SUBROUTINE MULTPY (FINAL ,AMATX ,START ,MAXAI ,NEQNS ,NWMTL )
C*********************************************************************
C
C *** TO EVALUATE PRODUCT OF B TIMES RR AND STORE RESULT IN TT
C
C*********************************************************************
      DIMENSION FINAL(1) ,AMATX(1) ,START(1) ,MAXAI(1)
C
```

```
      IF(NWMTL.GT.NEQNS) GO TO 20
      DO 10 IEQNS=1,NEQNS
   10 FINAL(IEQNS)=AMATX(IEQNS)*START(IEQNS)
      RETURN
C
   20 DO 40 IEQNS=1,NEQNS
   40 FINAL(IEQNS)=0.0
      DO 100 IEQNS=1,NEQNS
      LOWER=MAXAI(IEQNS)
      KUPER=MAXAI(IEQNS+1)-1
      JEQNS=IEQNS+1
      TERMI=START(IEQNS)
      DO 100 ICOLM=LOWER,KUPER
      JEQNS=JEQNS-1
  100 FINAL(JEQNS)=FINAL(JEQNS)+AMATX(ICOLM)*TERMI
      IF(NEQNS.EQ.1) RETURN
      DO 200 IEQNS=2,NEQNS
      LOWER=MAXAI(IEQNS)+1
      KUPER=MAXAI(IEQNS+1)-1
      IF(KUPER-LOWER) 200,210,210
  210 JEQNS=IEQNS
      SUMAA=0.0
      DO 220 ICOLM=LOWER,KUPER
      JEQNS=JEQNS-1
  220 SUMAA=SUMAA+AMATX(ICOLM)*START(JEQNS)
      FINAL(IEQNS)=FINAL(IEQNS)+SUMAA
  200 CONTINUE
      RETURN
      END
```

11.5.14 Subroutine REDBAK

This routine[9] solves the equations after the matrix is decomposed (into the form LDL^T) using forward and backward substitution.

```
      SUBROUTINE REDBAK (STIFF ,FORCE ,MAXAI ,NEQNS )
C*********************************************************************
C
C ****  TO REDUCE AND BACK-SUBSTITUTE ITERRATION VECTORS
C
C*********************************************************************
      DIMENSION STIFF(1) ,FORCE(1) ,MAXAI(1)
C
      DO 400 IEQNS=1,NEQNS
      LOWER =MAXAI(IEQNS)+1
      KUPER=MAXAI(IEQNS+1)-1
      IF(KUPER-LOWER) 400,410,410
  410 JEQNS=IEQNS
      SUMCC=0.0
      DO 420 ICOLM=LOWER,KUPER
      JEQNS=JEQNS-1
  420 SUMCC=SUMCC+STIFF(ICOLM)*FORCE(JEQNS)
      FORCE(IEQNS)=FORCE(IEQNS)-SUMCC
  400 CONTINUE
C
      DO 480 IEQNS=1,NEQNS
      KMAXA=MAXAI(IEQNS)
  480 FORCE(IEQNS)=FORCE(IEQNS)/STIFF(KMAXA)
      IF(NEQNS.EQ.1) RETURN
      JEQNS=NEQNS
      DO 500 IEQNS=2,NEQNS
      LOWER=MAXAI(JEQNS)+1
      KUPER=MAXAI(JEQNS+1)-1
```

```
      IF(KUPER-LOWER) 500,510,510
  510 KEQNS=JEQNS
      DO 520 ICOLM=LOWER,KUPER
      KEQNS=KEQNS-1
  520 FORCE(KEQNS)=FORCE(KEQNS)-STIFF(ICOLM)*FORCE(JEQNS)
  500 JEQNS=JEQNS-1
      RETURN
      END
```

11.5.15 Subroutine RESEPL

This routine evaluates the internal force vector for elasto-plastic materials. (See Section 7.8.7.)

```
      SUBROUTINE RESEPL (COORD ,DISPL ,EFFST ,ELOAD ,EPSTN ,IITER ,
     .                   INTGR ,LEQNS ,LNODS ,MATNO ,NCRIT ,NDIME ,
     .                   NDOFN ,NELEM ,NGAUS ,NLAPS ,NMATS ,NNODE ,
     .                   NPOIN ,NSTRE ,NTYPE ,POSGP ,PROPS ,RESID ,
     .                   STRAG ,STRIN ,STRSG ,WEIGP ,IPRED ,ISTEP )
C***********************************************************************
C
C *** EVALUATES RESIDUAL FORCES
C
C***********************************************************************
      DIMENSION COORD(NPOIN,1),DERIV(2,9),DMATX(4, 4),AVECT(4),MATNO(1),
     .          PROPS(NMATS,1),DLCOD(2,9),BMATX(4,18),DEVIA(4),DISPL(1),
     .          LNODS(NELEM,1),GPCOD(2,9),DJACM(2, 2),STRAN(4),POSGP(1),
     .          ELOAD(NELEM,1),CARTD(2,9),SHAPE(   9),STRES(4),WEIGP(1),
     .          STRIN(    4,1),ELCOD(2,9),SIGMA(   4),SGTOT(4),EFFST(1),
     .          STRSG(    4,1),ELDIS(2,9),DESIG(   4),DVECT(4),EPSTN(1),
     .          STRAG(    4,1),RESID(   1),LEQNS(18,1),INTGR(1)
      TWOPI=6.283185307179586
      NEVAB=NNODE*NDOFN
      NTOTV=NPOIN*NDOFN
      NSTR1=4
      DO 530 IELEM=1,NELEM
      IF(INTGR(IELEM).EQ.2.AND.IITER.GT.1.AND.IPRED.EQ.1) GO TO 530
      DO 540 IEVAB=1,NEVAB
  540 ELOAD(IELEM,IEVAB)=0.0
  530 CONTINUE
      DO 510 ITOTV=1,NTOTV
  510 RESID(ITOTV)=0.0
      KGAUS=0
      DO 20 IELEM=1,NELEM
      IF(INTGR(IELEM).EQ.2.AND.IITER.GT.1.AND.IPRED.EQ.1) GO TO 20
      LPROP=MATNO(IELEM)
      YOUNG=PROPS(LPROP,1)
      POISS=PROPS(LPROP,2)
      THICK=PROPS(LPROP,3)
      UNIAX=PROPS(LPROP,6)
      HARDS=PROPS(LPROP,7)
      FRICT=PROPS(LPROP,8)
      FRICT=FRICT*0.017453292
      IF(NCRIT.EQ.3) UNIAX=UNIAX*COS(FRICT)
      IF(NCRIT.EQ.4) UNIAX=6.0*UNIAX*COS(FRICT)/
     .                     (1.73205080757*(3.0-SIN(FRICT)))
C
C*** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      IPOSN=0
      DO 30 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
```

```
      DO 30 IDIME=1,NDIME
      IPOSN=IPOSN+1
      NPOSN=LEQNS(IPOSN,IELEM)
      IF(NPOSN.EQ.0) DISPT=0.
      IF(NPOSN.NE.0) DISPT=DISPL(NPOSN)
      DLCOD(IDIME,INODE)=COORD(LNODE,IDIME)+DISPT
      ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
   30 ELDIS(IDIME,INODE)=DISPT
      CALL MODPS (DMATX,LPROP,NMATS,NSTRE,NTYPE,PROPS)
      KGASP=0
      DO 40 IGAUS=1,NGAUS
      DO 40 JGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      ETASP=POSGP(JGAUS)
      KGAUS=KGAUS+1
      KGASP=KGASP+1
      CALL SFR2   (DERIV,NNODE,SHAPE,EXISP,ETASP)
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,
     .             IELEM,KGASP,NNODE,SHAPE)
      CALL JACOBD (CARTD,DLCOD,DJACM,NDIME,NLAPS,NNODE)
      DVOLU=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF(NTYPE.EQ.3) DVOLU=DVOLU*TWOPI*GPCOD(1,KGASP)
      IF(NTYPE.EQ.1) DVOLU=DVOLU*THICK
      CALL BLARGE (BMATX,CARTD,DJACM,DLCOD,GPCOD,
     .             KGASP,NLAPS,NNODE,NTYPE,SHAPE)
C
C
      CALL LINGNL (CARTD,DJACM,DMATX,ELDIS,GPCOD,KGASP,
     .             KGAUS,NDOFN,NLAPS,NNODE,NSTRE,NTYPE,
     .             POISS,SHAPE,STRAN,STRES,STRAG)
      DO 560 ISTR1=1,NSTR1
  560 STRAG(ISTR1,KGAUS)=STRAG(ISTR1,KGAUS)+STRAN(ISTR1)
      IF(ISTEP.GT.1.AND.IITER.GT.1) GO TO 160
      DO 170 ISTR1=1,NSTR1
  170 STRES(ISTR1)=STRES(ISTR1)+STRIN(ISTR1,KGAUS)
  160 CONTINUE
      PREYS=UNIAX+EPSTN(KGAUS)*HARDS
      DO 150 ISTR1=1,NSTR1
      DESIG(ISTR1)=STRES(ISTR1)
  150 SIGMA(ISTR1)=STRSG(ISTR1,KGAUS)+STRES(ISTR1)
      IF(NLAPS.EQ.2.OR.NLAPS.EQ.0) GO TO 60
      CALL INVAR  (DEVIA,LPROP,NCRIT,NMATS,PROPS,SINT3,STEFF,
     .             SIGMA,THETA,VARJ2,YIELD)
      ESPRE=EFFST(KGAUS)-PREYS
      IF(ESPRE.GE.0.0) GO TO 50
      ESCUR=YIELD-PREYS
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=ESCUR/(YIELD-EFFST(KGAUS))
      GO TO 70
   50 ESCUR=YIELD-EFFST(KGAUS)
      IF(ESCUR.LE.0.0) GO TO 60
      RFACT=1.0
   70 MSTEP=ESCUR*8.0/UNIAX+1.0
      IF(MSTEP.GT.10) MSTEP=10
      ASTEP=MSTEP
      REDUC=1.0-RFACT
      DO 80 ISTR1=1,NSTR1
      SGTOT(ISTR1)=STRSG(ISTR1,KGAUS)+REDUC*STRES(ISTR1)
   80 STRES(ISTR1)=RFACT*STRES(ISTR1)/ASTEP
      DO 90 JSTEP=1,MSTEP
      CALL INVAR  (DEVIA,LPROP,NCRIT,NMATS,PROPS,SINT3,STEFF,
     .             SGTOT,THETA,VARJ2,YIELD)
      CALL YIELDF (AVECT,DEVIA,FRICT,NCRIT,SINT3,STEFF,
     .             THETA,VARJ2)
      CALL FLOWPL (AVECT,ABETA,DVECT,HARDS,NTYPE,POISS,YOUNG)
```

```
      AGASH=0.0
      DO 100 ISTR1=1,NSTR1
  100 AGASH=AGASH+AVECT(ISTR1)*STRES(ISTR1)
      DLAMD=AGASH*ABETA
      IF(DLAMD.LT.0.0) DLAMD=0.0
      BGASH=0.0
      DO 110 ISTR1=1,NSTR1
      BGASH=BGASH+AVECT(ISTR1)*SGTOT(ISTR1)
  110 SGTOT(ISTR1)=SGTOT(ISTR1)+STRES(ISTR1)-DLAMD*DVECT(ISTR1)
      EPSTN(KGAUS)=EPSTN(KGAUS)+DLAMD*BGASH/YIELD
   90 CONTINUE
      CALL INVAR  (DEVIA,LPROP,NCRIT,NMATS,PROPS,SINT3,STEFF,
     .             SGTOT,THETA,VARJ2,YIELD)
      CURYS=UNIAX+EPSTN(KGAUS)*HARDS
      BRING=1.0
      IF(YIELD.GT.CURYS) BRING=CURYS/YIELD
      DO 130 ISTR1=1,NSTR1
  130 STRSG(ISTR1,KGAUS)=BRING*SGTOT(ISTR1)
      EFFST(KGAUS)=BRING*YIELD
C*** ALTERNATIVE LOCATION OF STRESS REDUCTION LOOP TERMINATION CARD
C  90 CONTINUE
C***
      GO TO 190
   60 DO 180 ISTR1=1,NSTR1
  180 STRSG(ISTR1,KGAUS)=STRSG(ISTR1,KGAUS)+DESIG(ISTR1)
      EFFST(KGAUS)=YIELD
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE
C    ELEMENT NODES
  190 MGASH=0
      DO 140 INODE=1,NNODE
      DO 140 IDOFN=1,NDOFN
      MGASH=MGASH+1
      DO 140 ISTRE=1,NSTRE
  140 ELOAD(IELEM,MGASH)=ELOAD(IELEM,MGASH)+BMATX(ISTRE,MGASH)*
     .STRSG(ISTRE,KGAUS)*DVOLU
   40 CONTINUE
   20 CONTINUE
      DO 500 IELEM=1,NELEM
      DO 500 IEVAB=1,NEVAB
      LMVEB=LEQNS(IEVAB,IELEM)
      IF(LMVEB.EQ.0) GO TO 550
      RESID(LMVEB)=RESID(LMVEB)+ELOAD(IELEM,IEVAB)
  550 CONTINUE
  500 CONTINUE
      RETURN
      END
```

11.6 Examples

11.6.1 Spherical shell example

Some of the capabilities[10] of the program MIXDYN are explained by analysing some simple problems. The spherical shell problem described[11,12] in Chapter 10 is again solved for the following cases:

(i) Elastic small deformation (all implicit elements)
(ii) Elastic geometrically nonlinear (all implicit elements)
(iii) Elasto-plastic small deformation (all implicit elements)

(iv) Elastic small deformation (all explicit elements)
(v) Elastic geometrically nonlinear (all explicit elements)
(vi) Elasto-plastic small deformation (all explicit elements)

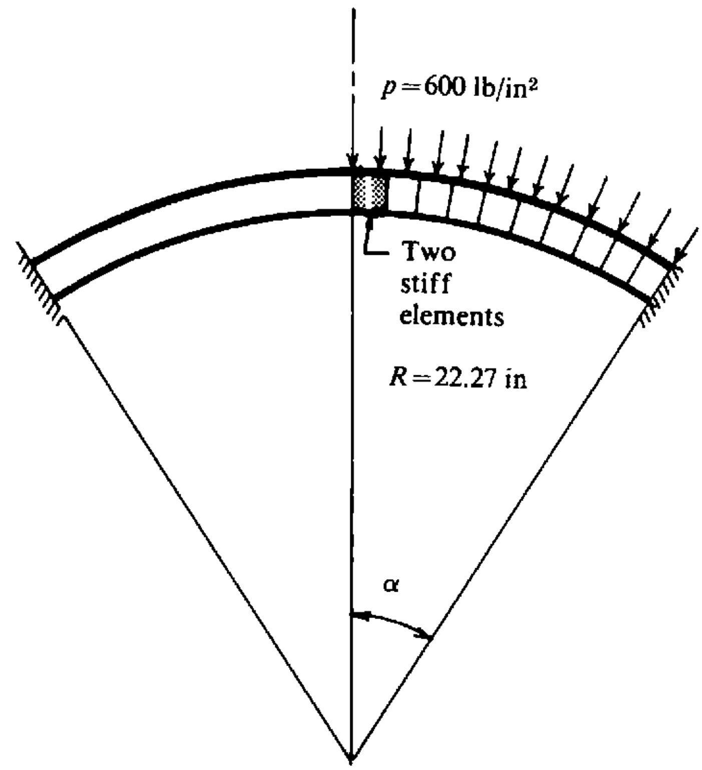


Fig. 11.4 Modified spherical shell example with stiff elements.

To demonstrate the capabilities of program MIXDYN we also solve a slightly modified version of the spherical shell example. Two stiff and dense elements are added to the finite element mesh at the crown as shown in Fig. 11.4. The stiff elements have the following properties:

Elastic modulus	E	$= 0.105 \times 10^9$ lb/in^2
poisson's ratio	ν	$= 0.3$
mass density	ρ	$= 0.780 \times 10^{-3}$ lb.sec^2/in^4
yield stress	σ_0	$= 0.5 \times 10^5$ lb/in^2

The following modified shell examples are also analysed:

(vii) Elasto-plastic small deformations (all implicit elements)
(viii) Elasto-plastic small deformations (all explicit elements)
(ix) Elasto-plastic small deformations (stiff elements are implicit elements, the remaining elements are explicit).

The highest and lowest eigenvalues are evaluated for both the original and the modified spherical shells. For the original spherical shell the fundamental period is 0.547×10^{-3} sec and the smallest time period is 1.380×10^{-6}

sec. For the modified spherical shell the fundamental period T_f is 0.592×10^{-3} sec and the smallest time period T_h is 0.776×10^{-6} sec. Thus the addition of the stiff elements does not significantly change the largest period but it does change the smallest period quite dramatically. For an accurate solution based on implicit time integration the time step length Δt is taken as $T_f/100 \simeq 0.6 \times 10^{-5}$ sec for both the original and the modified spherical shell. For a stable and accurate solution based on explicit time integration the time step length $\Delta t \leqslant T_h/\pi$ which is 0.25×10^{-6} sec for the modified spherical shell or 0.40×10^{-6} sec for the original spherical shell. Thus the addition of two stiff elements reduces the critical time step length to 1/1.6 of the original critical time step length. Hence the explicit analysis becomes more expensive. However, if the stiff elements are taken as implicit elements in case (ix) for implicit–explicit analysis, then the critical time step is governed by the remaining explicit elements so that the time step must be less than or equal to 0.40×10^{-6} sec.

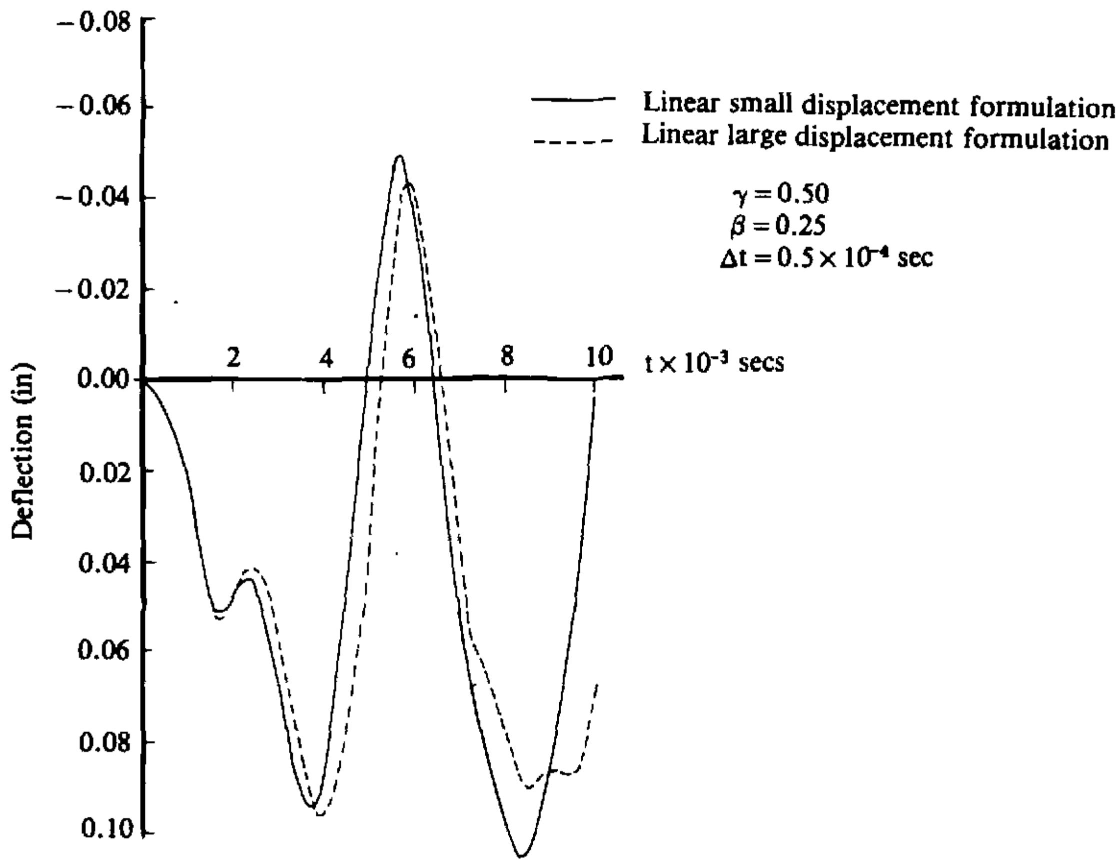


Fig. 11.5(a) Spherical shell results. Cases (i), (ii), (iv) and (v).

Figure 11.5(a) compares the response of the elastic analyses with small and large deformations.* The results are similar to the results obtained using DYNPAK. The response with the large deformation gives a time period which is elongated.

* Note that the implicit and explicit results overlap.

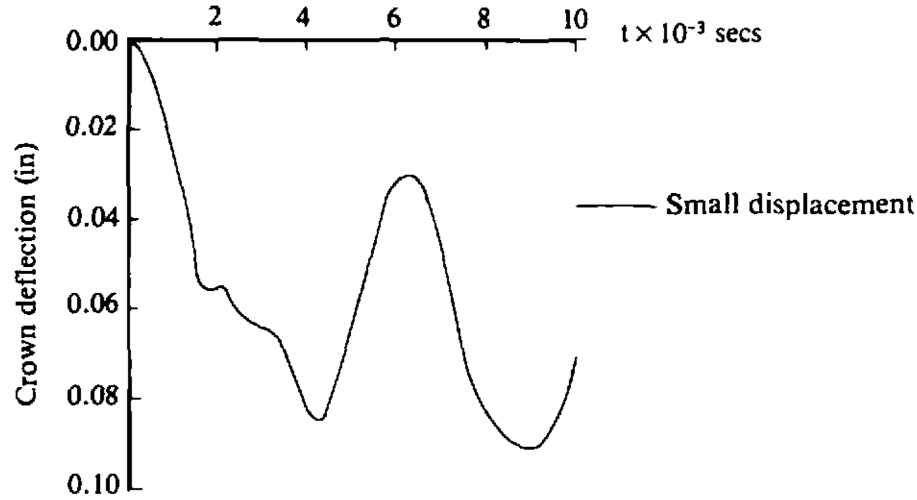
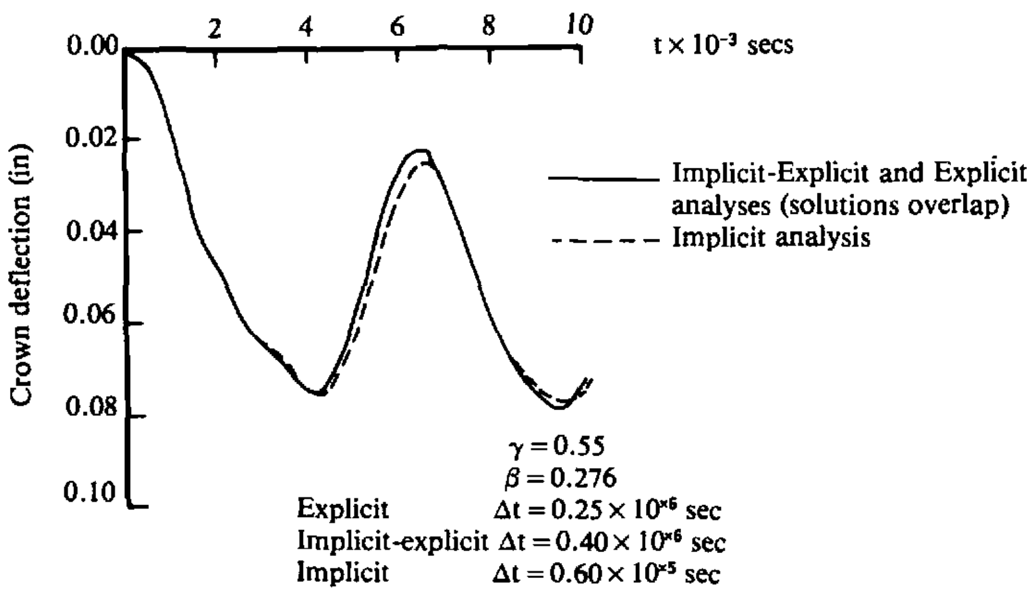


Fig. 11.5(b) Spherical shell results. Cases (iii) and (vi).

Figure 11.5(b) illustrates the elasto-plastic small deformation response. The time periods are elongated with the inclusion of plasticity effects.

In Fig. 11.5(c) the results for the problem with the stiff element are presented with explicit, implicit and mixed explicit–implicit analysis (cases (vii)–(ix)). The execution times and results are compared. The relative computer times are:

(i)	all elements considered as explicit	– 120.0 sec
(ii)	stiff elements as implicit and rest explicit	– 80.8 sec
(iii)	all elements considered as implicit	– 16.4 sec



(a) Comparison of explicit, implicit-explicit and implicit time integration schemes

Fig. 11.5(c) Spherical shell results. Cases (vii)–(ix).

This shows that by representing the stiff elements implicitly computer time can be saved. The analysis in which all elements are treated implicitly gives the lowest execution time for this small problem. However, with increasing problem size (and band width) the solution time for an implicit solution increases very rapidly because of the large core requirement and the increased number of computer operations.

Finally it should be noted that Hughes has recently shown how the implicit–explicit schemes may be used in a more general context where there are, for example, nonsymmetric stiffness matrices involved or an implicit–explicit dynamic relaxation solution is required.[13]

11.7 Problems

11.1 Repeat Problems 10.1–10.4 using program MIXDYN. Use fully explicit, fully implicit and mixed implicit/explicit meshes.

11.8 References

1. Belytschko, T. and Mullen, R., Mesh partitions of explicit–implicit time integration, In: *Formulations and Computational Algorithms in Finite Element Analysis*, Ed. K. J. Bathe *et al.*, MIT Press (1977).
2. Belytschko, T. and Mullen, R., Stability of explicit-implicit mesh partitions in time integration, *Int. J. Num. Meth. Engng.* **12,** 1575–1586 (1978).
3. Belytschko, T., Yen, H. J. and Mullen, R., Mixed methods for time integration, *Proc. Int. Conf. on Finite Elements in Non-linear Mechanics* (*FENOMECH 78*), Univ. Stuttgart, Germany (Sept. 1978).
4. Hughes, T. J. R. and Liu, W. K., Implicit-explicit finite elements in transient analysis: stability theory, *J. Appl. Mech.* **45,** 371–374 (1978).
5. Hughes, T. J. R. and Liu, W. K., Implicit-explicit finite elements in transient analysis: implementation and numerical examples, *J. Appl. Mech.* **45,** 375–378 (1978).
6. Hughes, T. J. R., Pister, K. S. and Taylor, R. L., Implicit-explicit finite elements in nonlinear transient analysis, *Comp. Meth. Appl. Mech. Engng.* **17/18,** 159–182 (1979).
7. Park, K. C., Felippa, C. A. and Deruntz, H. A., Stabilization of staggered solution procedures for fluid-structure interaction analysis, In: *Computational Methods for Fluid-Structure Interaction Problems*, Ed. T. Belytschko and T. L. Geers, ASME Applied Mechanics Symposia Series, AMD, **26,** 94–124 (1977).
8. Park, K. C., Partitioned transient analysis procedures for coupled field problems, to be published *J. Appl. Mech.* (1980).
9. Bathe, K. J. and Wilson, E. L., *Numerical Methods in Finite Element Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey (1976).
10. Paul, D. K. and Hinton, E., User guide and report on program MIXDYN for implicit-explicit transient dynamic analysis, *Research Report*, University College of Swansea (1980).
11. Nagarajan, S. and Popov, E. P., Elastic-plastic dynamic analysis of axisymmetric solids, *Computers and Structures*, **4,** 1117–1134 (1974).

12. BATHE, K. J. and OZDEMIR, H., Elastic-plastic large deformation static and dynamic analysis, *Computers and Structures*, **6**, 81–90 (1976).
13. HUGHES, T. J. R., Implicit–explicit finite element techniques for symmetric and nonsymmetric systems, *Proc. Int. Conf. Numerical Methods for Nonlinear Problems*, Swansea, 127–139, Pineridge Press, Swansea, U.K. (1980).

Chapter 12
Alternative formulations and further applications

12.1 Introduction

Throughout this text we have considered several specific elasto-plastic material problems and, apart from Chapter 3, treatment has been limited to the use of elasto-plastic quasi-static incremental theory or an elasto-visco-plastic formulation. These theories and the application areas of solids and plates form, undoubtedly, the area of most interest and importance in non-linear material analysis and it is for this reason that they have been chosen for study in this text. However, other topics and applications of possibly equal importance have had to be omitted for reasons of space and it is the aim of this chapter to indicate to the reader some areas for future studies. The developments which will be discussed can be categorised into the following classes:

- *Further applications.* The elasto-plastic and elasto-viscoplastic theories described earlier in this text can be extended to cover some alternative structural forms. Of prime importance in this area is the analysis of both thick and thin three-dimensional shell structures and the main changes necessary to the corresponding linear elastic finite element process relate to expressing the yield criterion in terms of the appropriate stress resultants.
- *Alternative material models.* The behaviour of some engineering materials may not be adequately described by the yield criteria presented in Chapter 7. This is particularly true of soils, rocks and concrete, since these materials, for example, have a limited tensile strength which is not accurately reflected in either the Mohr–Coulomb or Drucker–Prager failure laws. For such materials appropriate failure criteria must be developed. Additionally for soils the assumption of associated plasticity leads to excessive dilatency necessitating alternative formulations for accurate material modelling.
- *Further problem classes.* Many physical situations exist which are governed by nonlinear equation systems which are not suitable for solution by the techniques described so far in the text. One such

example is the time dependent deformations which take place during a metal forming process. In this application the elastic strains are negligible compared with the plastic components and therefore the stress increments can no longer be expressed by use of (8.15).

For dynamic situations, coupled media problems frequently have to be solved. This may involve a fluid/structure interaction problem of the seismic analysis of water retaining structures or the impulsive loading of a nuclear containment vessel together with the coolant fluid. All the above problems may be complicated by further nonlinear behaviour due to gross geometrical deformations.

- *Improved numerical techniques.* Since nonlinear solution processes are necessarily expensive with regard to computational time, any savings which can be made in this area are of prime importance. Developments in this area include improved nonlinear equation solution techniques and self-adaptive schemes for optimisation of the finite element mesh and load incrementation. A further enhancement is the use of substructuring techniques to separate elastic and elasto-plastic regions leading ultimately to coupled boundary integral/finite element solutions.

In this chapter we explore the above developments (and others) in more detail and provide the reader with references for future study. Many of the subroutines presented earlier in the text can be employed (possibly in a modified form) in the development of computer codes for these further applications. Therefore the role of each subroutine presented is summarised and its location in the text also listed.

12.2 List of subroutines

In this section we record details of each subroutine that has been presented in this text. This library of subroutines can be employed to develop computer codes for the further applications discussed later in this chapter. The section of the chapter in which the subroutine is presented is recorded and the codes in which it is used are also indicated, employing the following program names:

One-dimensional applications

QUITER	Solution of quasiharmonic problems by direct iteration (Chapter 3).
QUNEWT	Solution of quasiharmonic problems by the Newton–Raphson process (Chapter 3).
NONLAS	Solution of nonlinear elastic problems (Chapter 3),
ELPLAS	Solution of elasto-plastic problems (Chapter 3).
UNVIS	Solution of elasto-viscoplastic problems (Chapter 4).
TIMOSH	Solution of elasto-plastic nonlayered Timoshenko beams (Chapter 5).
TIMLAY	Solution of elasto-plastic layered Timoshenko beams (Chapter 5).

Two-dimensional applications

PLANET	Elasto-plastic analysis of plane stress, plane strain and axi-symmetric solids (Chapter 7).
VISCOUNT	Elasto-viscoplastic analysis of plane stress, plane strain and axisymmetric solids (Chapter 8).
MINDLIN	Elasto-plastic analysis of nonlayered Mindlin plates (Chapter 9).
MINDLAY	Elasto-plastic analysis of layered Mindlin plates (Chapter 9).
DYNPAK	Elasto-plastic transient dynamic analysis of two dimensional solids (Chapter 10).
MIXDYN	Implicit–explicit elasto-viscoplastic transient dynamic analysis of two dimensional solids (Chapter 11).

12.2.1 Subroutines for one-dimensional applications

ASSEMB — Section 3.4.2 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Assembles the element contributions to form the global stiffness matrix and global load vector. (Simple equation solver).

ASTIF1 — Section 3.10.1 (QUNEWT)
Formulates the stiffness matrix for each element according to (2.25) and (2.29) for the solution of one dimensional quasi-harmonic problems by the Newton Raphson method.

BAKSUB — Section 3.4.4 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Performs the backsubstitution phase of the Gaussian reduction process. (Simple equation solver).

BEAM — Section 5.4.5 (TIMOSH)
The master routine for elasto-plastic nonlayered Timoshenko beam program TIMOSH.

BEML — Section 5.5.5 (TIMLAY)
The master routine for elasto-plastic layered Timoshenko beam program TIMLAY.

CONUND — Section 3.10.3 (QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Monitors convergence of the nonlinear solution process based on the residual forces according to (3.27).

CONVP — Section 4.9 (UNVIS)
Monitors convergence to steady state conditions according to (4.41) for one-dimensional elasto-viscoplastic problems.

DATA — Section 3.2 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Data input subroutine for one-dimensional applications.

GREDUC Section 3.4.3 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Undertakes equation elimination by Gaussian reduction. (Simple equation solver).

INCLOD Section 3.7 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Controls the incrementing of the applied loads for one-dimensional applications (modified for viscoplastic problems in Section 4.10).

INCVP Section 4.8 (UNVIS)
Evaluates quantities at the end of the time step and the equilibrium correction terms for one-dimensional elasto-viscoplastic problems.

INITAL Section 3.6 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Initialises to zero some arrays used by other subroutines for one-dimensional applications.

MONITR Section 3.9.2 (QUITER)
Monitors convergence of the direct iteration process for one-dimensional quasiharmonic problems.

NONAL Section 3.3 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Controls the nonlinear solution process according to the value of NALGO specified, for one-dimensional applications.

REFOR1 Section 3.10.2 (QUNEWT)
Evaluates the 'equivalent nodal forces' according to (3.26) for one-dimensional quasiharmonic problems. (Newton Raphson solution).

REFOR2 Section 3.11.2 (NONLAS)
Evaluates the equivalent nodal forces according to (3.32) for one-dimensional nonlinear elastic problems.

REFOR3 Section 3.12.2 (ELPLAS)
Evaluates the equivalent nodal forces for one-dimensional elasto-plastic problems.

REFORB Section 5.4.5 (TIMOSH)
Evaluates the residual forces for a nonlayered elasto-plastic Timoshenko beam.

RFORBL Section 5.5.5 (TIMLAY)
Evaluates the residual forces for a layered elasto-plastic Timoshenko beam.

RESOLV Section 3.4.5 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Undertakes reduction of the R.H.S. terms for equation resolution (Simple equation solver).

RESULT Section 3.5 (QUITER, QUNEWT, NONLAS, ELPLAS, TIMOSH, TIMLAY)
Outputs the results for one-dimensional applications.

STIFF1 Section 3.9.1 (QUITER)
Formulates the stiffness matrix for each element according to (2.25) for the solution of one-dimensional quasiharmonic problems by direct iteration.

STIFBL Section 5.5.5 (TIMLAY)
Evaluates the elasto-plastic stiffness matrix for each element for the solution of layered Timoshenko beams.

STIFFB Section 5.4.5 (TIMOSH)
Formulates the elasto-plastic stiffness matrix for each element for the solution of nonlayered Timoshenko beams.

STIFF2 Section 3.11.1 (NONLAS)
Formulates the stiffness matrix for each element according to (2.33) for nonlinear elastic one-dimensional problems.

STIFF3 Section 3.12.1 (ELPLAS)
Formulates the stiffness matrix for each element according to either (2.38) or (2.43) for one-dimensional elasto-plastic problems.

STUNVP Section 4.7 (UNVIS)
Formulates the stiffness matrix for each element in turn for one-dimensional elasto-viscoplastic applications.

UNDIM Section 3.8 (QUITER, QUNEWT, NONLAS, ELPLAS)
The main or master segment for one-dimensional nonlinear problems. See Fig. 3.1 for the small changes in the different applications.

UNVISC Section 4.11 (UNVIS)
The main or master segment for one-dimensional visco-plastic problems.

12.2.2 Subroutines for two-dimensional applications

ADDBAN Section 11.5.3 (MIXDYN)
Generates the global matrix from the element stiffness matrices.

ADDRES Section 11.5.4 (MIXDYN)
Addresses the diagonal term of a matrix.

ALGOR Section 6.5.2 (PLANET, VISCOUNT, MINDLIN, MINDLAY)
Controls the nonlinear solution process according to the value of NALGO specified, for two-dimensional applications.

BLARGE Section 10.6.3 (DYNPAK, MIXDYN)
Evaluates the strain matrix $\boldsymbol{B}$ for small and large deformation.

BMATPB Section 6.4.8 (MINDLIN)
Evaluates the strain matrix, $\boldsymbol{B}$, for plate bending problems.

BMATPS	Section 6.4.7 (PLANET, VISCOUNT) Evaluates the strain matrix, $\boldsymbol{B}$, for plane and axisymmetric situations.
CHECK1	Section 6.4.13 (PLANET, VISCOUNT, MINDLIN, MINDLAY) Scrutinises the problem control parameters for possible errors (two-dimensional applications).
CHECK2	Section 6.4.15 (PLANET, VISCOUNT, MINDLIN, MINDLAY) Checks the geometric data, boundary conditions and material properties for possible errors (two-dimensional applications).
COLMHT	Section 11.5.5 (MIXDYN) Evaluates the height of column above the diagonal of a matrix from the known addresses of diagonal terms.
CONTOL	Section 10.6.4 (DYNPAK, MIXDYN) Reads control data for dynamic dimensioning and also checks the dimension limits.
CONVER	Section 6.5.4 (PLANET) Monitors convergence of the nonlinear solution iteration process for two-dimensional applications.
CONVMP	Section 9.5.3 (MINDLIN, MINDLAY) Checks for convergence of solution of elasto-plastic layered and nonlayered Mindlin plates.
DBE	Section 6.4.11 (PLANET, VISCOUNT) Forms the matrix product $\boldsymbol{DB}$.
DECOMP	Section 11.5.6 (MIXDYN) Decomposes positive definite matrix into $\boldsymbol{LDL}^T$.
DEPMPA	Section 9.6.4 (MINDLAY) Sets up the layered discretisation for the layered elasto-plastic Mindlin plate.
DIMEN	Section 7.8.1 (PLANET, VISCOUNT) Presets the value of variables associated with dynamic dimensioning.
DIMMP	Section 9.5.4 (MINDLIN, MINDLAY) Sets up dynamic dimensions in programs MINDLIN and MINDLAY for the elasto-plastic analysis of layered and nonlayered plates.
DINTOB	Section 11.5.7 (MIXDYN) Multiplies the modulus and strain matrices to give $\boldsymbol{DB}$.
DYNPAK	Section 10.6.2 (DYNPAK) Organises the explicit viscoplastic transient dynamic analysis.
ECHO	Section 6.4.14 (PLANET, VISCOUNT, MINDLIN, MINDLAY)

Echoes the remaining data after input data errors have been diagnosed.

EXPLIT Section 10.6.5 (DYNPAK)
Carries out explicit time integration.

FEAM Section 9.6.2 (MINDLAY)
Organising routine for the elasto-plastic analysis of layered Mindlin plates.

FEMP Section 9.5.2 (MINDLIN)
Organising routine for the elasto-plastic analysis of nonlayered Mindlin plates.

FIXITY Section 10.6.6 (DYNPAK)
Boundary conditions are inserted.

FLOWMP Section 9.5.5 (MINDLIN, MINDLAY)
Determines $\partial F/\partial \boldsymbol{\sigma}_f$ (i.e. yield function derivatives) for elasto-plastic layered and nonlayered Mindlin plates.

FLOWPL Section 7.8.4.2 (PLANET, MIXDYN)
Determines the vector $\boldsymbol{d}_D$ for elasto-plastic analysis.

FLOWVP Section 8.9 (VISCOUNT, DYNPAK)
Determines the viscoplastic strain rate for each Gauss point according to (8.7).

FRONT Section 6.4.12 (PLANET, VISCOUNT, MINDLIN, MINDLAY)
Performs element assembly and equation solution by the frontal method. Contains a facility for efficient resolution of equations.

FUNCTA Section 10.6.8 (DYNPAK, MIXDYN)
Interpolates acceleration ordinate at Δt intervals.

FUNCTS Section 10.6.9 (DYNPAK, MIXDYN)
Evaluates factor for Heaviside and Harmonic time function at Δt apart.

GAUSSQ Section 6.4.2 (PLANET, VISCOUNT, MINDLIN, MINDLAY, DYNPAK, MIXDYN)
Evaluates the sampling point positions and weighing factors for numerical integration by Gauss quadrature.

GEOMST Section 11.5.8 (MIXDYN)
Evaluates the stress stiffness matrix.

GRADMP Section 9.5.6 (MINDLIN)
Evaluates the total displacement and rotation derivatives ($\partial w/\partial x$, $\partial w/\partial y$, $\partial \theta_x/\partial x$, $\partial \theta_x/\partial y$, $\partial \theta_y/\partial x$, $\partial \theta_y/\partial y$).

GSTIFF Section 11.5.9 (MIXDYN)
Evaluates the global stiffness matrix in compacted profile form.

IMPEXP Section 11.5.10 (MIXDYN)
Sets the constants of integration and evaluates partial effective load vector.

INCREM — Section 6.5.3 (PLANET, VISCOUNT, MINDLIN, MINDLAY)
Controls the incrementing of the applied loads for two-dimensional applications.

INPUT — Section 6.5.1 (PLANET, VISCOUNT, MINDLIN, MINDLAY)
Data input subroutine for two-dimensional applications.

INPUTD — Section 10.6.10 (DYNPAK, MIXDYN)
Data input subroutine. Reads the mesh data, properties etc

INTIME — Section 10.6.11 (DYNPAK, MIXDYN)
Reads the data necessary for time integration.

INVAR — Section 7.8.3 (PLANET, VISCOUNT, DYNPAK, MIXDYN)
Evaluates the effective stress level at a given point for monitoring plastic yielding.

INVERT — Section 8.7.3 (VISCOUNT)
This subroutine determines the inverse of any arbitrary square matrix.

INVMP — Section 9.5.7 (MINDLIN)
Evaluates the Mindlin plate stress resultant invariants for nonlayered plates.

ITRATE — Section 11.5.11 (MIXDYN)
Evaluates the total effective load and iterates until convergence is reached.

JACOBD — Section 10.6.13 (DYNPAK, MIXDYN)
Evaluates the deformation Jacobian matrix.

JACOB2 — Section 6.4.4 (PLANET, VISCOUNT, MINDLIN, MINDLAY, DYNPAK, MIXDYN)
Evaluates the Jacobian matrix, its inverse and the Cartesian derivatives of the element shape functions for two-dimensional applications.

LAYMPA — Section 9.6.5 (MINDLAY)
Evaluates the matrix of flexural rigidities and the matrix of shear rigidities for the layered elastoplastic Mindlin plate.

LINEAR — Section 7.8.6 (PLANET, MIXDYN)
Determines the stresses from given displacements assuming linear elastic behaviour.

LINGNL — Section 10.6.14 (DYNPAK, MIXDYN)
Evaluates the linear stresses for small and large deformation analysis.

LINKIN — Section 11.5.12 (MIXDYN)
This routine links with the profile solver.

LOADPB — Section 6.4.6 (MINDLIN, MINDLAY)
Evaluates the consistent nodal forces for plate bending problems.

LOADPL Section 10.6.15 (DYNPAK, MIXDYN)
Generates the load vector.

LOADPS Section 6.4.5 (PLANET, VISCOUNT)
Evaluates the consistent nodal forces due to gravity and distributed edge loads for two-dimensional problems.

LUMASS Section 10.6.16 (DYNPAK, MIXDYN)
Generates the consistent mass matrix for implicit elements and special lumped mass matrix for explicit elements.

MDMPA Section 9.6.6 (MINDLAY)
Evaluates the constitutive matrices for use in layered Mindlin plate analysis.

MINDPB Section 9.5.8 (MINDLIN, MINDLAY)
Reads additional input data for elasto-plastic, layered and nonlayered Mindlin plates.

MIXDYN Section 11.5.2 (MIXDYN)
Organises implicit/explicit transient dynamic program.

MODPB Section 6.4.10 (MINDLIN)
Evaluates the $\boldsymbol{D}$ matrix for plate bending applications.

MODPS Section 6.4.9 (PLANET, VISCOUNT, DYNPAK, MIXDYN)
Evaluates the $\boldsymbol{D}$ matrix for plane and axisymmetric situations.

MULTPY Section 11.5.13 (MIXDYN)
Multiplies square matrix to a vector or vector to a vector.

NODEXY Section 6.4.1 (PLANET, VISCOUNT, MINDLIN, MINDLAY)
Interpolates the coordinates of midside nodes for elements with straight sides. This routine is modified in MINDLIN and MINDLAY where a hierarchical formulation is adopted for the ninth node. (See Section 9.5).

NODXYR Section 10.6.18 (DYNPAK, MIXDYN)
Evaluates the midside node of elements. In case of axisymmetric problems if (R, Θ) coordinates are read r, z coordinates are evaluated within it.

OUTDYN Section 10.6.19 (DYNPAK, MIXDYN)
Writes the output on output file and stress and displacement histories of required Gauss points and nodes respectively on specified tapes.

OUTMP Section 9.5.10 (MINDLIN)
Outputs displacements, reactions and Gauss point stress resultants for elasto-plastic nonlayered Mindlin plates.

OUTMPA Section 9.6.7 (MINDLAY)
Outputs displacements, reactions and Gauss point layer stresses for elasto-plastic layered Mindlin plates.

OUTPUT Section 7.8.8 (PLANET, VISCOUNT)
Outputs the results for two-dimensional problems at specified intervals.

PLAST Section 7.8.9 (PLANET)
The main or master segment for two-dimensional elasto-plastic applications.

PREVOS Section 10.6.20 (DYNPAK, MIXDYN)
Reads the initial force and stresses.

REDBAK Section 11.5.14 (MIXDYN)
Solves equations after matrix decomposition, using forward and backward substitution.

RESEPL Section 11.5.15 (MIXDYN)
Evaluates the internal force for different yield criteria in the implicit explicit program.

RESMP Section 9.5.11 (MINDLIN)
Evaluates the internal nodal forces

$$p = \int_{\Omega} \mathbf{B}_f{}^T \boldsymbol{\sigma}_f \, d\Omega + \int_{\Omega} \mathbf{B}_s{}^T \boldsymbol{\sigma}_s \, d\Omega$$

for the stress resultants $\boldsymbol{\sigma}_f$ and $\boldsymbol{\sigma}_s$ for elasto-plastic, non-layered Mindlin plates.

RESMPA Section 9.6.8 (MINDLAY)
Evaluates the residual force vector for layered elasto-plastic Mindlin plates.

RESIDU Section 7.8.7 (PLANET)
Evaluates the nodal forces which are statically equivalent to the stress field satisfying elasto-plastic conditions.

RESVPL Section 10.6.21 (DYNPAK)
Evaluates the internal forces for different yield criteria in the explicit transient dynamic program.

SFR2 Section 6.4.3 (PLANET, VISCOUNT, MINDLIN, MINDLAY, DYNPAK, MIXDYN)
Evaluates the element shape functions and their local derivatives for 4, 8 and 9 node isoparametric quadrilateral elements. SFR2 is modified in MINDLIN and MINDLAY to allow for a hierarchical representation for the 9th central node.

STEADY Section 8.12 (VISCOUNT)
Monitors convergence to steady state conditions for two-dimensional elasto-viscoplastic problems.

STEPVP Section 8.8 (VISCOUNT)
Evaluates quantities, such as stresses and viscoplastic strains, at the end of each time step of a viscoplastic solution.

STIFFP Section 7.8.5 (PLANET)

	Evaluates the stiffness matrix for each element for elasto-plastic problems employing either $\boldsymbol{D}$ or $\boldsymbol{D}_{ep}$ as appropriate.
STIFMP	Section 9.5.13 (MINDLIN) Evaluates the stiffness matrices for nonlayered elasto-plastic Mindlin plate elements.
STIFVP	Section 8.7.1 (VISCOUNT) Evaluates the stiffness matrix for each element in turn for two-dimensional elasto-viscoplastic applications.
STIMPA	Section 9.6.9 (MINDLAY) Evaluates the stiffness matrices for layered elasto-plastic Mindlin plate elements.
STRESS	Section 8.10 (VISCOUNT) Evaluates the increment in stress occurring during a timestep of a viscoplastic analysis according to (8.20).
STRMP	Section 9.5.14 (MINDLIN) Evaluates stress resultants $[M_x, M_y, M_{xy}, Q_x, Q_y]^T$ for elasto-plastic nonlayered Mindlin plates.
STRMPA	Section 9.6.10 (MINDLAY) Evaluates the stresses $[\sigma_x, \sigma_y, \tau_{xy}, \tau_{xz}, \tau_{yz}]^T$ for elasto-plastic layered Mindlin plates at each layer and each Gauss point.
SUBMP	Section 9.5.15 (MINDLIN, MINDLAY) Carries out matrix multiplications in elasto-plastic layered and nonlayered Mindlin plates.
TANGVP	Section 8.7.2 (VISCOUNT) Evaluates the $\boldsymbol{D}^n$ matrix for viscoplastic analysis by implicit time stepping schemes.
VISCO	Section 8.13 (VISCOUNT) The main or master segment for two-dimensional elasto-viscoplastic applications.
VZERO	Section 9.5.16 (MINDLIN, MINDLAY) Zeroes a vector in elasto-plastic layered and nonlayered Mindlin plates.
YIELDF	Section 7.8.4.1 (PLANET, VISCOUNT, MIXDYN, DYNPAK) Determines the flow vector $\boldsymbol{a}$ for plastic and viscoplastic applications. (Amended in Section 10.6.22 for dynamic transient problems).
ZERO	Section 7.8.2 (PLANET, VISCOUNT) Sets to zero the contents of several arrays employed in the programs. (Modified for viscoplastic applications in Section 8.11).
ZEROMP	Section 9.5.16 (MINDLIN, MINDLAY) Zeroes various arrays in elasto-plastic layered and nonlayered Mindlin plate programs.

12.3 Alternative material models

The plastic behaviour of most solids is adequately described by the four yield criteria presented in Chapter 7; namely the Tresca, Von Mises, Mohr–Coulomb and Drucker–Prager yield surfaces. However, for some engineering materials, notably concrete, rocks and soils, some modifications must be made to the above criteria or new yield surfaces postulated if an accurate prediction of the material response is required.

For soils, the Mohr–Coulomb and Drucker–Prager criteria suffer from two deficiencies. Firstly, the assumption of an associated flow rule leads to excessive dilatency and secondly it is seen from Fig. 7.4 that both models imply that the material can support an unlimited hydrostatic compression. These deficiencies can be removed by use of the so-called *critical state model*, which assumes that the yield surface comprises two distinct parts.[1–3] The surface is shown plotted in terms of deviatoric σ_d and hydrostatic stress, σ_s, in Fig. 12.1. In the subcritical region yielding is stable due to strain hardening of the material whilst the supercritical region exhibits strain softening so that this portion of the yield surface forms a failure criterion.

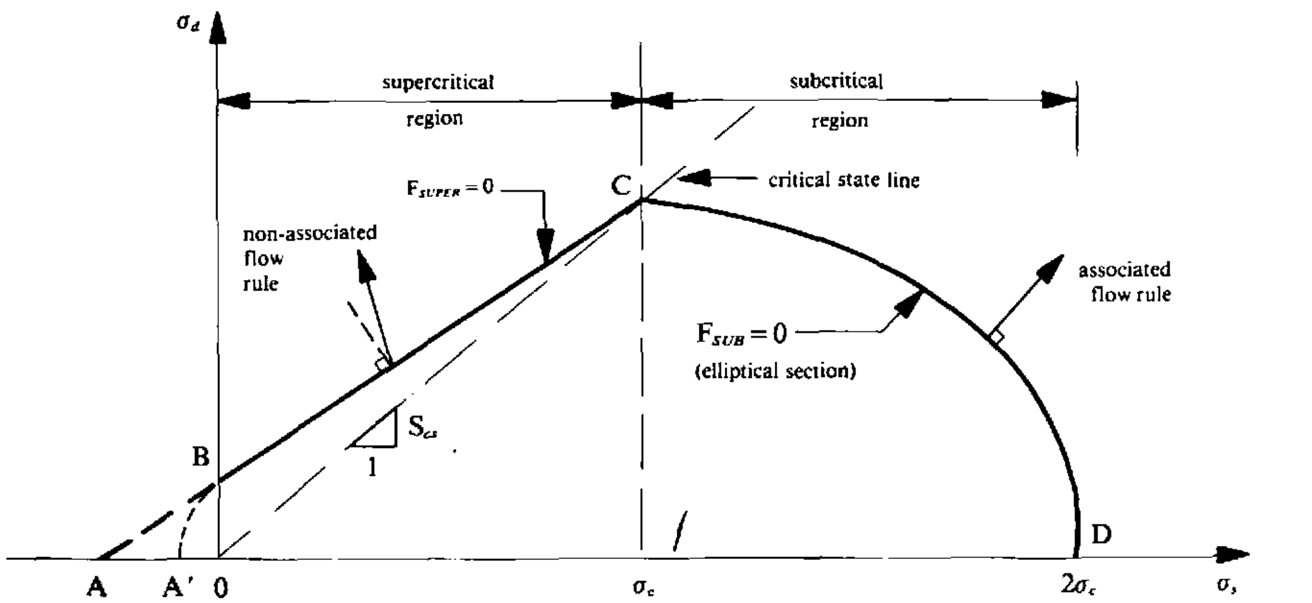


Fig. 12.1 Critical state model for the behaviour of soil, [$\sigma_d = |\sigma_1 - \sigma_3|$, $\sigma_s = \frac{1}{2}(\sigma_1 + \sigma_3)$].

A nonassociative flow rule is adopted in the supercritical region and the conical yield surface implied in Fig. 12.1 may be circular or hexagonal in form corresponding to a Mohr–Coulomb behaviour. In the subcritical region, the two most common shapes for the so-called cap is a log spiral or an ellipse and an associated flow rule is assumed to be obeyed. The yield surface can be expressed in the form

$$F_{\text{SUPER}} = \sigma_d - 2\sin\phi\,\sigma_s - 2c\cos\phi = 0$$

$$F_{\text{SUB}} = \frac{\sigma_d^2 - S_{cs}^2\,\sigma_s(2\sigma_c - \sigma_s)}{\sigma_d + S_{cs}\,\sigma_c} = 0, \tag{12.1}$$

in which S_{cs} is the slope of the critical state line.

In the tensile zone, various options are open for modelling the limited tensile strength of the soil. The curved line BA' can be employed or, more simply the vertical intercept OB (implying zero tensile strength) may be assumed. Complete details of the critical state model for soils can be found in Refs. 1–3 including its application to the numerical solution of practical problems.

The Mohr–Coulomb and Drucker–Prager criteria exhibit the same deficiencies for modelling concrete behaviour as occur in the case of soils. In particular they overestimate the tensile strength of the material and also allow the material to support an unlimited hydrostatic compression. Many models have been proposed to more accurately predict the behaviour of concrete; a review of which can be found in Ref. 4.

The most common method of predicting the tensile behaviour of concrete (and rocks) is by use of the *no-tension model* (or limited tension model).[5] In this, the tensile principal stresses are monitored throughout the structure and as soon as the value at any point exceeds the specified limiting tensile strength of the concrete, the material is assumed to crack in a plane normal to the principal direction. The tensile stress must then be reduced to zero by evaluating its nodal force equivalent and regarding these as residual forces to be applied and redistributed in an iterative process. Should the crack close on load reversal a frictional behaviour between the surfaces of the crack can be modelled. It is worth recording that the numerical stability of such solution processes is relatively poor since on initiation of tensile cracking the existing stress must be eliminated by redistribution, whereas for elasto-plastic problems, yielding merely necessitates that the existing stress level be maintained.

An example of this type of analysis is illustrated in Fig. 12.2 where a cylindrical prestressed concrete reactor vessel is shown. The geometry of the vessel, together with the location of the prestressing system is indicated and the finite element mesh employed in solution is also shown. The concrete is assumed to behave as a limited tension material and the steel components as a Von Mises elasto-plastic solid. The effects of prestressing are included as an initial stress system and the vessel is incrementally loaded by a progressively increasing internal pressure. Figure 12.3 shows the vertical deflection of the centre point of the end slab with increasing load and good agreement is observed with both the experimental results and numerical analysis of Ref. 6. The zones of tensile cracking are shown in Fig. 12.4 for various applied pressure values and again good agreement with the results of Ref. 6 is evident.

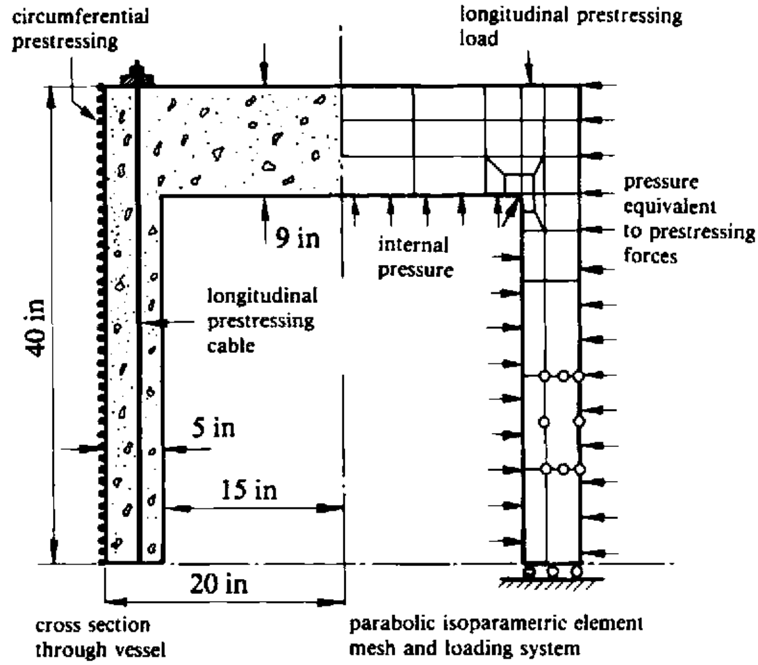


Fig. 12.2 Finite element idealisation of a prestressed concrete reactor vessel by quadratic isoparametric elements.

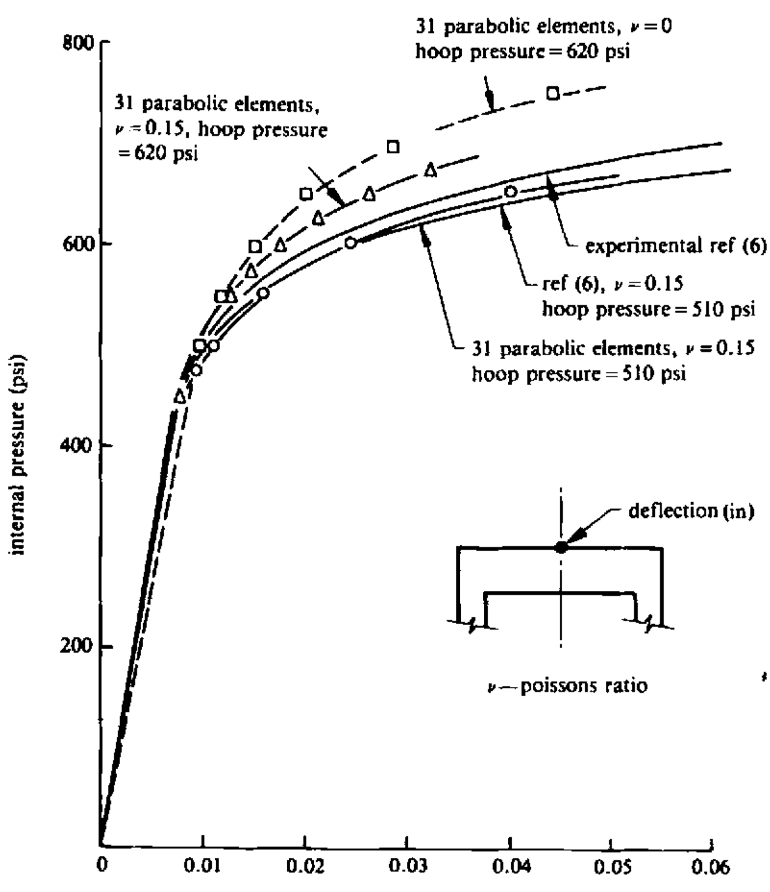


Fig. 12.3 Load/deflection curves for the vessel of Fig. 12.2 failing in slab flexural mode.

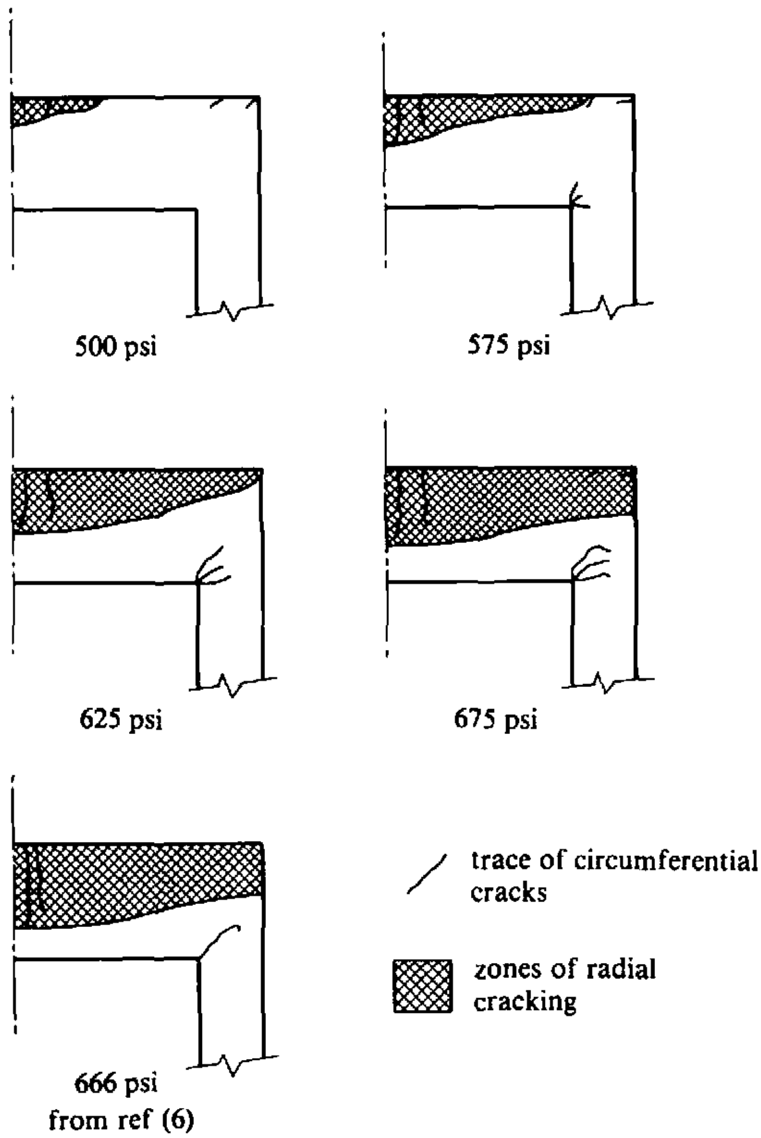


Fig. 12.4 Zones of tensile cracking for the vessel of Fig. 12.2 failing in slab flexural mode.

For predicting the compressive behaviour of concrete as well as the tensile response many failure surfaces have been proposed and a typical model is illustrated in Fig. 12.5. In addition to a brittle behaviour in tension, the model allows a viscoplastic range of behaviour before material failure. For further details the reader is directed to Ref. 4.

A final approach to concrete behaviour which is worthy of mention is afforded by the so-called *endochronic theory* pioneered by Valanis(7,8) and generalised to concrete structures by Bazant.(9,10) To account for the strain history dependence of materials (in addition to their strain rate dependence) the concept of *intrinsic time* z is introduced which is related to the Newtonian time scale, t according to

$$dz^2 = \alpha^2(d\zeta^2 + \beta^2 dt^2), \tag{12.2}$$

where $d\zeta$ is effectively a measure of the deformation path length, β is a material parameter and a depends on $\dot{\zeta}$. Bazant has generalised the endochronic model to account for inelastic dilatancy, hydrostatic and shear compaction and fracture behaviour.[10]

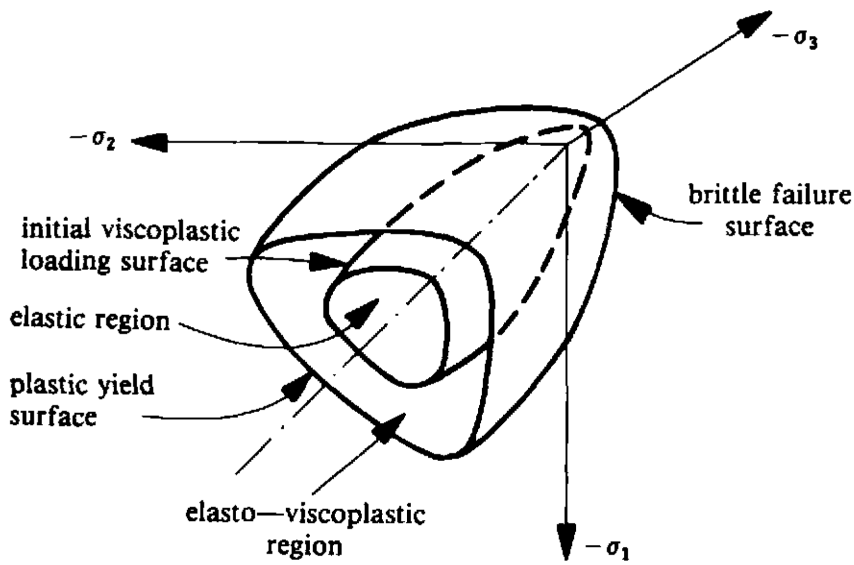


Fig. 12.5 Typical yield and failure surfaces for concrete.

12.4 Further applications

12.4.1 Flow problems

In this class of problem we are concerned with the continuing viscous flow of materials under steady state conditions. Typical examples include the extrusion of material through a die and flow of lubricating muds in oil drilling applications. In each case the problem is characterized by the fact that the elastic strains are negligible in comparison to the plastic components. For this reason, the viscoplastic numerical process described in Chapter 8 is unsuitable, since the increment of stress occurring during a timestep was based on the *elastic* strain increment according to (8.15). Thus an alternative formulation is clearly necessary and in fact a considerable simplification is achieved if the elastic components of strain are neglected in solution.[11]

The plastic strain rate, $\dot{\boldsymbol{\epsilon}}_{vp}$, which is now assumed to be the total strain rate, $\dot{\boldsymbol{\epsilon}}$, is given from (8.7) to be

$$\dot{\boldsymbol{\epsilon}} = \dot{\boldsymbol{\epsilon}}_{vp} = \gamma\langle\Phi(F)\rangle \boldsymbol{a}, \tag{12.3}$$

and we recall that $\boldsymbol{a}$ is the flow vector defined by (7.42), Φ is an appropriate flow function (given for example by (8.8) or (8.9)) and γ is a fluidity parameter. For the particular case of a Von Mises yield surface we have from (7.11) that

$$F(\sigma,\kappa) = \sqrt{3}(J_2')^{1/2} - \sigma_Y(\kappa), \tag{12.4}$$

where J_2' is the second deviatoric stress invariant and σ_Y is the uniaxial yield stress of the material which may be a function of the strain hardening parameter κ. Substituting from (12.4) into (12.3), and using (7.42) to express $\boldsymbol{a}$, results in

$$\dot{\boldsymbol{\epsilon}} = \gamma\langle\Phi(\sqrt{3}(J_2')^{1/2}-\sigma_Y)\rangle\,\sqrt{3}/2(J_2')^{1/2}\,\boldsymbol{\sigma}' = \boldsymbol{\Gamma}(\boldsymbol{\sigma}')\boldsymbol{\sigma}', \tag{12.5}$$

in which $\boldsymbol{\sigma}'$ are the deviatoric stresses and $\boldsymbol{\Gamma}(\boldsymbol{\sigma}')$ is a symmetric viscoplastic compliance matrix whose form can be explicitly determined on prescription of the appropriate flow function Φ. Thus a relationship has been established between the total strain rate and the deviatoric stresses.

The strain rate can be expressed in terms of the displacement velocities $\boldsymbol{v}$ by taking the differential form of the standard strain/displacement relationship, to give

$$\dot{\boldsymbol{\epsilon}} = \boldsymbol{B}\boldsymbol{v}. \tag{12.6}$$

We assume, as for the viscoplastic case of Chapter 8, that the flow velocities are sufficiently slow to neglect inertia effects and that the following standard static equilibrium equations therefore hold.

$$\int_V \boldsymbol{B}^T\boldsymbol{\sigma}\,dV+\boldsymbol{f} = 0, \tag{12.7}$$

in which $\boldsymbol{f}$ are the applied forces comprising body forces $\boldsymbol{b}$ and boundary tractions, $\boldsymbol{t}$. Thus a complete analogy exists between the above problem and the case of an elastic material in which the relationship between stress and strain is nonlinear according to

$$\boldsymbol{\sigma} = \boldsymbol{D}(\boldsymbol{\sigma})\boldsymbol{\epsilon}. \tag{12.8}$$

Table 12.1 Correspondence between small strain nonlinear elastic problems and viscoplastic flow situations

Small strain nonlinear elasticity	Flow problem
Displacements, $\mathbf{d}$	Velocities, $\boldsymbol{v}$
Stresses, $\boldsymbol{\sigma}$	Stresses, $\boldsymbol{\sigma}$
Strains, $\boldsymbol{\epsilon}$	Strain rates, $\dot{\boldsymbol{\epsilon}}$
Applied forces, $\boldsymbol{f}$	Applied forces, $\boldsymbol{f}$
Nonlinear elastic compliance matrix, $[\boldsymbol{D}(\boldsymbol{\sigma})]^{-1}$	Viscoplastic compliance matrix, $\Gamma(\boldsymbol{\sigma})$

This analogy is indicated in Table 12.1. Therefore flow problems, in which the elastic components of deformation are negligible, can be solved by use of a linear elastic computer code which includes a facility for dealing with a stress dependent $\boldsymbol{D}$ matrix. Obviously the steady state solution to the flow problem must be arrived at in an iterative manner and a similar procedure must be employed in the corresponding elastic solution. The simplest approach is to proceed by the method of direct iteration, as described in Chapters 2 and 3, and to base the value of the compliance matrix $\mathbf{\Gamma}$ on the current value of $\boldsymbol{\sigma}$. This solution procedure can be summarised as follows:

(1) From the stresses $\boldsymbol{\sigma}^n$ at iteration n evaluate the viscoplastic compliance matrix $\mathbf{\Gamma}(\boldsymbol{\sigma}^n) = \mathbf{\Gamma}^n$.
(2) Compute the element stiffness matrix of each element as

$$\int_V \boldsymbol{B}^T[\mathbf{\Gamma}^n]^{-1}\boldsymbol{B}\,dV$$

and also the consistent nodal applied forces, $\boldsymbol{f}^{(e)}$.
(3) Assemble and solve the stiffness equations to give the improved velocity estimate, $\boldsymbol{v}^{n+1}$.
(4) Compute the strain rates, $\dot{\boldsymbol{\epsilon}}^{n+1} = \boldsymbol{B}\boldsymbol{v}^{n+1}$.
(5) Compute the stresses, $\boldsymbol{\sigma}^{n+1} = [\mathbf{\Gamma}^n]^{-1}\dot{\boldsymbol{\epsilon}}^{n+1}$.
(6) Return to Step 1 and repeat the process until convergence takes place (i.e. $\boldsymbol{v}^{n+1} \approx \boldsymbol{v}^n$).

The procedure described above is most suitable when boundary and body forces produce the forcing action. For the case when the problem is defined in terms of prescribed boundary velocities the compliance matrix $\mathbf{\Gamma}$ must be expressed in terms of the current strain rate, $\dot{\boldsymbol{\epsilon}}$.[12]

For metal forming problems, the situation is complicated by the fact that the geometry of the deforming solid is continually varying throughout the process. For such problems the transient form of the flow equations must be used and an incremental procedure can be adopted by which the coordinates of the finite element mesh are sequentially updated during solution.[13]

It should be noted that no volumetric strain rate exists for some viscoplastic flow laws, as generally defined by (12.3), and this is indeed the case for the Von Mises criterion employed in (12.5). Consequently the viscoplastic compliance matrix $\mathbf{\Gamma}$ cannot be inverted as required by Step 2 above and the same numerical difficulties that exist in incompressible elastic problems are encountered. However these can be readily overcome by the use of *selective integration techniques* whereby the element stiffness matrix is separated into volumetric and deviatoric components.[14] The near singularity arising in the former term as incompressible behaviour is approached is then numerically removed by employing a low order Gaussian integration rule.

An important application of the above solution process is to the flow of non-Newtonian fluids, in which the material viscosity depends nonlinearly on the shear strain rate. Practical examples of such flow can be found in Refs. 15 and 16. Deviations from Newton's law of viscosity are best illustrated by means of flow curves and some of the most important cases are shown in Fig. 12.6. The effective stress, $\bar{\sigma}$, and effective strain rate, $\bar{\dot{\epsilon}}$, are defined by (7.12) and (7.22) respectively.

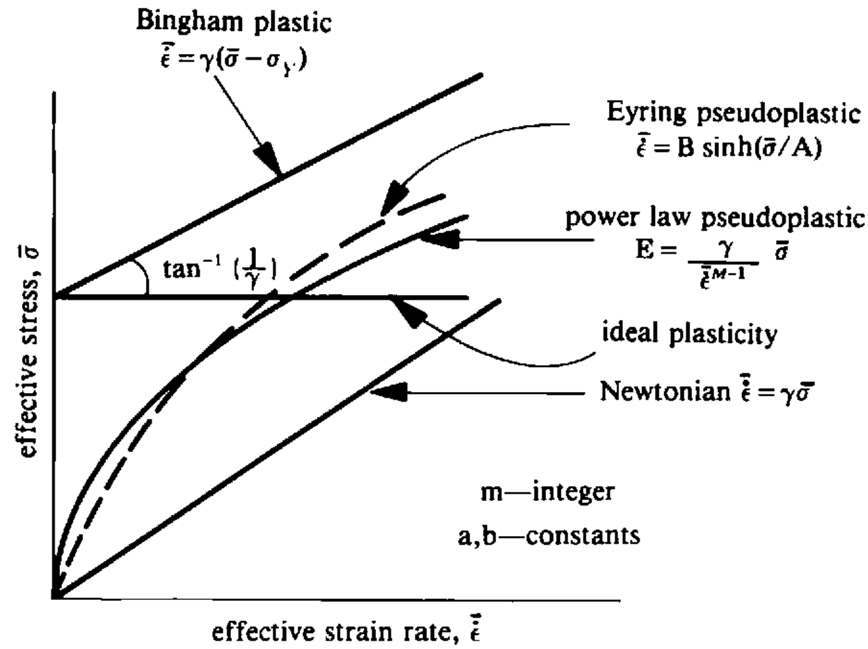


Fig. 12.6 Various flow curves for non-Newtonian fluids.

The Bingham fluid is seen to be a particular form of viscoplastic relation (12.3) or (12.5). Writing in terms of the effective stress and strain rate, (12.5) can be expressed as

$$\bar{\sigma} = \mu \bar{\dot{\epsilon}}, \tag{12.9}$$

where the apparent viscosity μ is given by

$$\frac{1}{\mu} = \frac{\sqrt{(3)}\gamma}{2(J_2')^{1/2}} \langle \Phi[(\sqrt{3})(J_2')^{1/2} - \sigma_Y] \rangle. \tag{12.10}$$

For the Bingham plastic we can write from the expression given in Fig. 12.6 and using (12.9) that

$$\mu = \frac{\bar{\dot{\epsilon}}/\gamma + \sigma_Y}{\bar{\dot{\epsilon}}}. \tag{12.11}$$

As $\gamma \to \infty$, ideal plasticity behaviour is approached resulting in

$$\mu = \frac{\sigma_Y}{\bar{\dot{\epsilon}}}. \tag{12.12}$$

Similarly for a Power Law pseudoplastic we have from Fig. 12.6

$$\mu = \frac{\bar{\dot{\epsilon}}^{M-1}}{\gamma}. \tag{12.13}$$

Thus for each case the problem again reduces to an elastic problem in which the shear modulus is dependent on the current strain rate and can be solved by use of the analogy indicated in Table 12.1. Solution can be achieved by use of the method of direct iteration or by the Newton–Raphson process described in Chapters 2 and 3.

As an example of viscous flow analysis[17] the problem of the flow of a Bingham fluid in a cylindrical annulus is illustrated in Fig. 12.7, where the geometry and finite element mesh employed are also indicated. Steady state flow is induced parallel to the axis of the cylinder by the application of an axial pressure gradient. The finite element velocity distributions obtained by a direct iteration solution scheme are shown in Fig. 12.8 for different values of the pressure gradient. The flow velocities are in good agreement with the theoretical solution of Ref. 18.

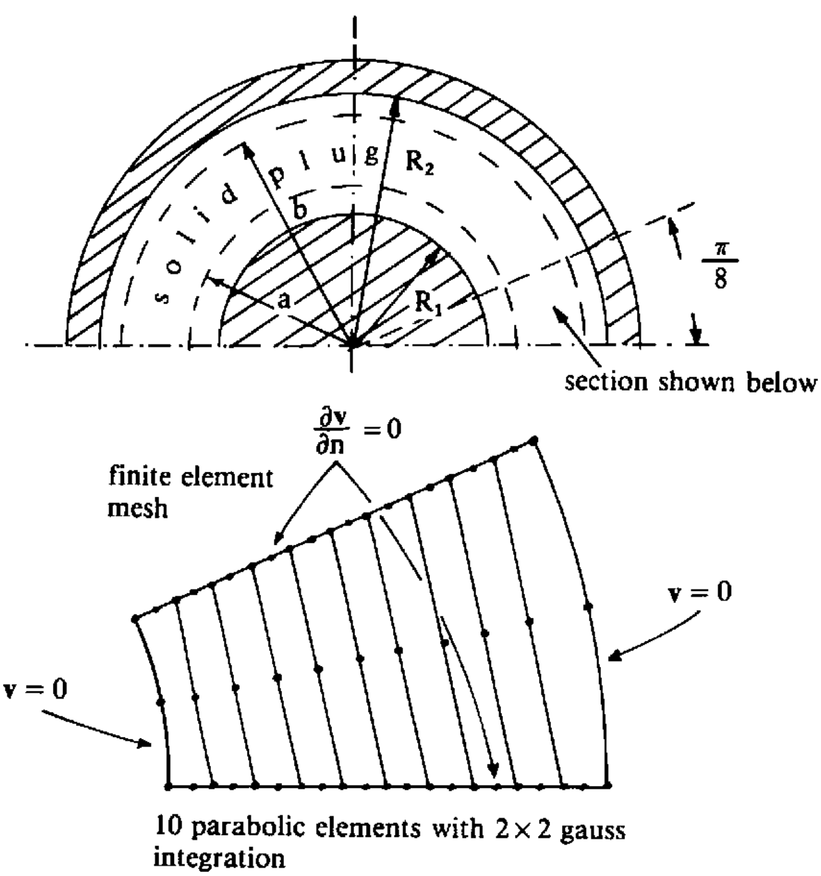


Fig. 12.7 Flow of Bingham fluid in an annulus under an axial pressure gradient showing finite element mesh idealisation.

12.4.2 Nonlinear fracture mechanics

A class of elasto-plastic problems which require special attention is that of crack propagation in ductile materials. Figure 12.9 illustrates the types of problem which demand solution and it is seen that a geometrical singularity exists at the crack tip. The numerical techniques presented in Chapter 7 allows the elasto-plastic stress field to be determined in the vicinity of the crack tip (for Modes I and II at least) but a criterion for propagation of the crack must be established in some way.

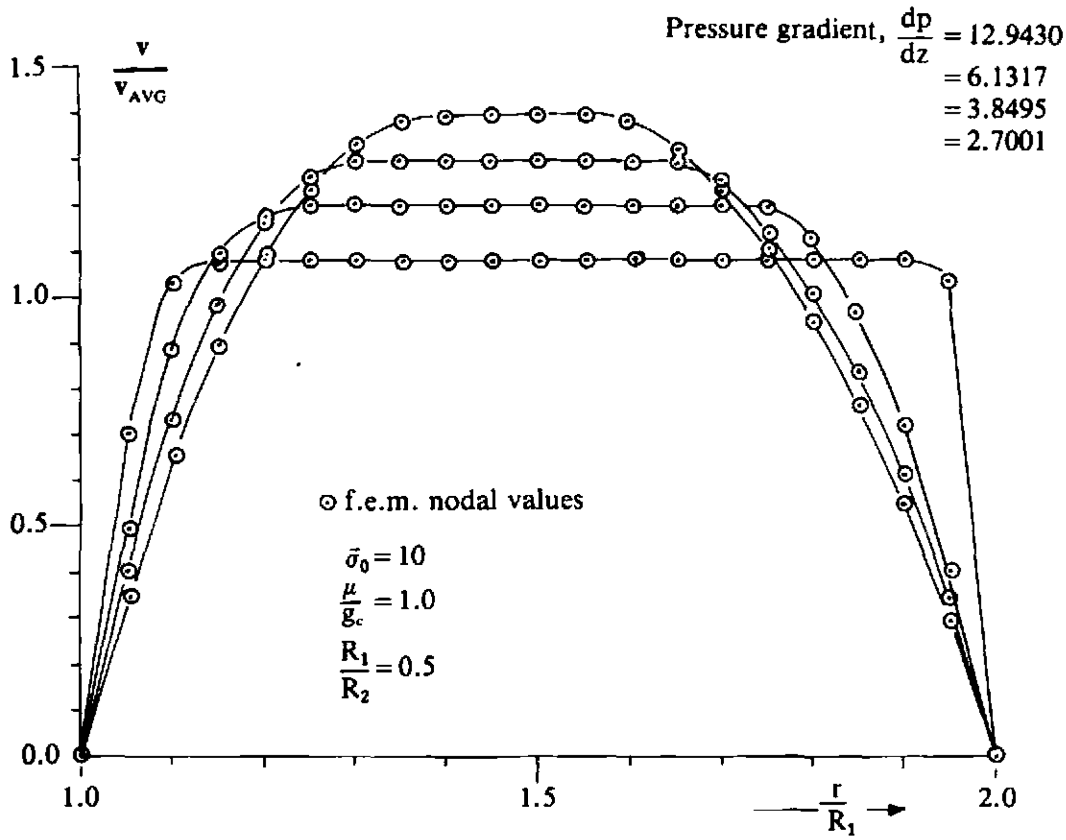


Fig. 12.8 Steady state velocity profile for the problem of Fig. 12.7 for various applied pressure gradients.

For linear elastic fracture problems crack advance can be monitored by specifying a critical value of a quantity, K, termed the *stress intensity factor** which characterises the stress field in the vicinity of the crack tip according to[20]

$$\sigma = Kf(\theta)/\sqrt{(2\pi r)} + \text{terms of order } r^0. \tag{12.14}$$

A separate K parameter exists for each fracture mode, designated by K_I, K_{II} and K_{III} respectively and they are functions only of geometry and loading conditions. A crack in any mode is then assumed to propagate when K attains a critical value K_c which is treated as a material parameter.

We now seek a similar criterion for elasto-plastic material behaviour. The most widely accepted principle in present use is the so-called *J contour integral* attributed to Rice[21] and which was originally formulated for non-linear elastic applications. The J integral is defined to be

$$J = \int_\Gamma \omega\, dy - T_i \frac{du_i}{dx} dS, \tag{12.15}$$

for a crack aligned in the x direction. Here Γ is any contour from the lower crack face leading anticlockwise around the crack tip to the upper face, S is the path length around this contour and $T_i du_i$ is the work contribution

* An excellent introduction to fracture mechanics is provided in Refs. 19 and 24.

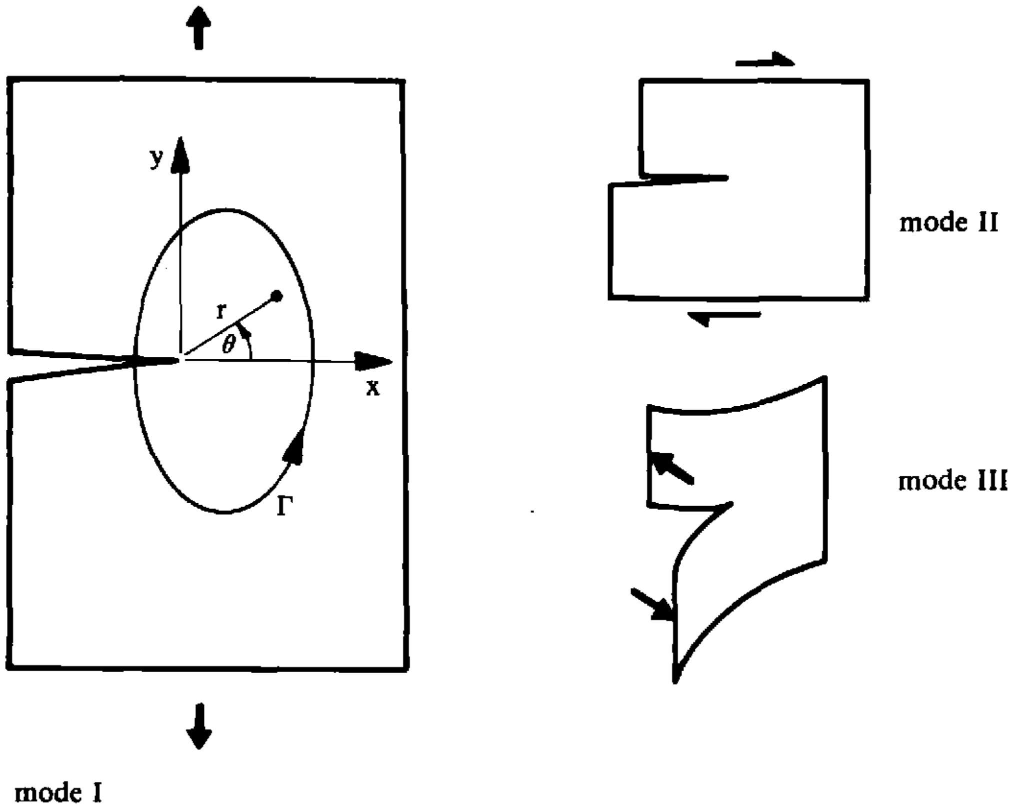


Fig. 12.9 Basic modes of fracture.

of traction components T_i on Γ moving through displacements du_i. The term ω is the strain energy density defined as

$$\omega = \int_0^{\epsilon} \sigma_{ij}\, d\epsilon_{ij}. \tag{12.16}$$

The J integral is independent of the choice of path Γ provided that the faces of the crack are stress free.

For Mode I opening in a strain-hardening *nonlinear elastic material* the near tip solution for the stress, strain and displacement can be shown to be of the form[22–24]

$$\sigma = C \frac{1}{r^{1/(N+1)}} \sigma(\theta)$$

$$\epsilon_p = C \frac{1}{r^{N/(N+1)}} \epsilon(\theta)$$

$$u = C r^{1/(N+1)} u(\theta), \tag{12.17}$$

where

$$C = \left(\frac{JE}{\sigma_Y{}^2 I}\right)^{1/(N+1)}. \tag{12.18}$$

The term N is a constant which measures the strain hardening of the material, E the elastic modulus, σ_Y the stress denoting the limit of linearity and I is a tabulated constant whose value depends on N.

For loading situations, nonlinear elastic behaviour is identical to that of a material obeying the laws of 'deformation' plasticity[25] in which the current stiffness is a function only of the current state of deformation and not of the loading path by which this condition has been reached. Furthermore for monotonic loading, experience indicates that there is no significant difference between solutions obtained by use of 'deformation' theories and the incremental theory adopted in Chapter 7. By this argument it is concluded that expressions (12.17) and (12.18) are applicable to elasto-plastic solids. Consequently crack propagation in elasto-plastic materials is governed by a critical value of the J integral.

One of the difficulties of numerical fracture studies is that a reasonably accurate prediction of the stress field in the vicinity of the crack tip is required. This is a computationally expensive process for elasto-plastic problems and in some instances economies can be made by use of special crack tip elements. For example, in Mode II deformation under plastic conditions, a shear strain singularity of order $1/r$ develops, which has been modelled by Levy *et al.*[26] by coalescing two nodes of a linear quadrilateral isoparametric element and treating their displacements independently. This approach has also been employed by Rice *et al.*[27]

12.4.3 Coupled-field problems

The transient analysis of many engineering systems involves the formulation of the semi-discrete coupled-field equations of motion which are then solved by a time-stepping procedure.[28] Coupled-field equations involving plasticity arise in the modelling of structure–fluid interaction, soil–fluid interaction, structure–structure interaction, etc. There are two main sources of difficulty in solving such problems:

(i) The isolated fields may display quite different response characteristics which may only be analysed efficiently by different time integration algorithms and/or different time steps.

(ii) Most engineering software has been developed for the treatment of single-field problems. The term 'partitioned transient analysis procedures' has been used to describe methods which allow the direct time integration of the entire equations to be performed by either sequential or parallel execution of single-field analyzers.

We have discussed partitioned procedures for structural dynamic problems in Chapter 11. We described an implicit–explicit partition through which meshes that exhibit high (low) frequency response characteristics are treated by implicit (explicit) integration formulae. Park[29] has recently extended the approach described in Chapter 11.

Park *et al.*[30] have studied implicit–implicit partitions in certain types of fluid-structure interaction problems. The solution of these coupled-field equations was obtained by a sequential execution of fluid and structural analyzers which gave rise to the term 'staggered solution procedures.'

Hughes[31] has summarised recent work on transient fluid-structure interaction problems. In particular he mentions work on procedures known as mixed, or arbitrary, Lagrangian–Eulerian methods.

In recent work on soil liquefaction problems, Zienkiewicz *et al.*[32] have devised a model which couples the soil and pore-fluid behaviour during earthquakes. Pore pressure build up and pore water migration are both accurately modelled.

Many other coupled-field problems involving elasto-plastic behaviour have been reported in the literature. It should however be emphasised that care should be taken in considering the stability of such schemes.

12.4.4 Elasto-plastic and geometrically nonlinear analyses of plates and shells

The linear and nonlinear finite element analysis of plates and shells has attracted much attention in the last decade. Two basic approaches have been adopted:

(i) *The classical procedure*

Here a plate or shell theory is used as a basis for the finite element formulation. Let us briefly summarise such an approach. We begin with the field equations of the three-dimensional theory and make various assumptions which lead to the plate or shell theory. In the reduction from three to two dimensions we include an analytical integration over the thickness. We then base our finite element discretisation process on the plate or shell theory. The surface geometry (in the case of shells) and the field variables are approximated using discrete nodal values and suitable interpolation functions. Integration of the various element stiffness and force terms is carried out over the reference surface. Stresses may then be obtained from the stress resultants. Examples of such an approach include the simple facet element and the many elements derived from classical thin plate theory, Mindlin/Reissner plate theory, shallow shell theory or even higher order shell theories.[33,34] There are very many examples of the application of the classical procedures in nonlinear finite element analysis of plates and shells. We include a brief sample in the list of references to this chapter.[35–38] For elasto-plastic problems many research workers express the yield function in terms of the stress resultants (cf. the non-layered approach in Chapter 9). For example, Crisfield[39–44] uses a modified Ilyushin yield criterion expressed in terms of the bending moments $[M_x, M_y, M_{xy}]^T$ and the membrane forces $[N_x, N_y, N_{xy}]^T$. To allow for the gradual spread of plasticity over the plate or shell thickness, a modified classical procedure may be adopted in which integration through the thickness is performed numerically during the finite element stiffness and force evaluation rather than analytically prior to the finite element discretisation. Gauss–Legendre, Lobatto and the mid-ordinate rules are frequently used for this purpose. To allow for geometrically nonlinear effects, total or updated Lagrangian approaches are adopted.[45–55]

(ii) *Ahmad and related elements*
Here isoparametric elements with independent rotational and displacement degrees of freedom are used. This concept originally introduced by Ahmad *et al.*[56] was later extended to allow for the linear analysis of thin as well as moderately thick shells by Zienkiewicz *et al.*[57] by the use of the reduced integration technique.*

Ahmad elements were originally developed because of the computational difficulties encountered in the use of the usual three-dimensional elements for the analysis of plates and shells. In the three-dimensional elements the stiffness coefficients corresponding to the transverse displacement degrees of freedom are very much larger than those corresponding to the longitudinal displacements. Erroneous strain energy corresponding to the normal stresses in the thickness direction are also introduced. Both of these difficulties are overcome in Ahmad elements. Normals to the plate or shell reference surface before deformation are assumed to remain straight but not necessarily normal to the reference surface after deformation. Furthermore, the normal stresses in the direction of the shell thickness are ignored and suitably modified constitutive equations are adopted.

Various nonlinear problems have been solved using Ahmad shell elements by Ramm[67], Krakeland[68], Bathe and Bolourchi[69] and others[70–73]. As in the modified classical procedures, to allow for the gradual spread of plasticity over the plate or shell thickness, numerical integration techniques are adopted. For geometrically nonlinear behaviour both total and updated

* The Mindlin plate elements described in Chapters 6 and 9 are simply plate versions of the Ahmad elements in which integration has been carried out analytically through the plate thickness. Much work on reduced and selective integration techniques[58–65] eventually led to the recognition that the use of selective integration techniques is equivalent to the use of a special type of mixed formulation.[66] Defects in the Ahmad elements have now been widely acknowledged and the use of the 9-node heterosis Mindlin plate element and the 16-node cubic Ahmad element are usually recommended. Other Ahmad/Mindlin $C(0)$ elements should be used with caution as they are known to give overstiff solutions for thin plates and shells and to develop mechanisms (zero energy modes) or near mechanisms (artificially low energy modes) when reduced or selective integration is used.

Lagrangian schemes have been used. Special techniques have been incorporated to allow for large rotations in the total Lagrangian formulations.[67–69]

The Ahmad shell concept has been developed further by its originator Irons with the introduction of the Semiloof element.[90] Irons adopted a convenient nodal configuration involving rotational degrees of freedom at 'Loof' nodes on the curved boundaries of the element. By imposing a series of constraints to eliminate transverse shear effects (reminiscent of the discrete Kirchhoff hypothesis), a highly effective thin shell element is obtained. Various research workers[74–76] have successfully extended this work into the nonlinear range.

Both classical and Ahmad procedures may be used as a basis for the nonlinear analysis of reinforced concrete plates and shells using the layering concept described in Chapter 9. Special constitutive relationships are required to represent the concrete and steel reinforcing bars are treated as a 'smeared' layer with uni-directional elasto-plastic properties. Much work has been completed in this area.[77–85]

Elasto-viscoplastic plates and shells are easily developed using the concepts described in Chapters 8 and 9.[86,87]

12.5 Equation solving techniques

12.5.1 Standard and modified Newton method

Before considering some alternative nonlinear solution procedures which may be used in elastoplastic finite element analysis we review the techniques described earlier.

As we have already seen, most elasto-plastic finite element programs are simply extensions of elastic finite element programs with linearised load increments. Some form of iterative procedure is usually adopted to dissipate the out-of-balance nodal forces.

The standard and variety of modified Newton methods were described earlier in Part I. Recall that the standard Newton method involves iterations in which

$$\boldsymbol{K}^{(i)}[\boldsymbol{d}^{(i+1)}-\boldsymbol{d}^{(i)}] = \boldsymbol{\psi}(\boldsymbol{d}^{(i)}), \qquad (12.19)^*$$

where $\boldsymbol{d}$ is the vector of nodal displacements and the equations $\boldsymbol{\psi}(\boldsymbol{d}) = \boldsymbol{0}$ express a force balance (internal forces = external forces; either for an increment of loading or for the whole applied load). The matrix $\boldsymbol{K}$ in the standard Newton method is the Jacobian of $\boldsymbol{\psi}$; which is the tangential stiffness matrix $\boldsymbol{K}_T = [\partial\boldsymbol{\psi}(\boldsymbol{d}^{(i)})/\partial\boldsymbol{d}]$ evaluated at the displacements described by $\boldsymbol{d}^{(i)}$.

The modified Newton method works with a variety of approximations to $\boldsymbol{K}$, the most simple of which is the initial elastic stiffness matrix $\boldsymbol{K}_0$ evaluated at the first iteration of the first load increment.

* The superscripts denote the iteration number.

We have adopted standard and modified Newton methods throughout this text as they are the most widely used approaches. Though they work well they do have certain disadvantages. The initial stiffness method is slow to converge in cases in which there is a high degree of nonlinearity. The modified Newton methods provide better convergence properties but they diverge during elastic unloading and they can lead to ill-conditioned or singular Jacobian matrices $\boldsymbol{K}$ near the limit load.

Newton methods are sometimes employed with a slight modification during an iteration in which

$$\boldsymbol{K}^{(i)}\Delta\boldsymbol{d}^{(i)} = \boldsymbol{\psi}^{(i)}, \tag{12.20}$$

and in which the new displacement vector is given as

$$\boldsymbol{d}^{(i+1)} = \boldsymbol{d}^{(i)}+\alpha^{(i)}\Delta\boldsymbol{d}^{(i)}, \tag{12.21}$$

where we could take $\alpha^{(i)}$ as much less than 1 for safety or more than 1 for more rapid convergence. Nayak[88] introduced an acceleration technique in which $\alpha^{(i)}$ is replaced by a diagonal matrix. Basu[89] later simplified this technique.

Although the modified Newton methods with fixed values of $\alpha^{(i)}$ is employed by certain analysts, it has been suggested[90] that we should reject it in favour of a modified Newton with a line search which involves finding a value of $\alpha^{(i)}$ which minimises the total potential energy $\pi(\boldsymbol{d}^{(i+1)})$ or the value of

$$Q = |[\boldsymbol{d}^{(i)}]^T\boldsymbol{\psi}(\boldsymbol{d}^{(i+1)})|. \tag{12.22}$$

12.5.2 Quasi-Newton method

Over the past twenty years there has been a rapid development of computer-oriented, sequential search methods in the fields of optimisation and mathematical programming. Of these techniques, the variable metric (Quasi–Newton) method and the method of conjugate gradients show the greatest potential in nonlinear finite element analysis.

The Quasi–Newton method was introduced to finite element computations by Matthies and Strang.[91] The main idea is to update the matrix $\boldsymbol{K}$ in a simple way after each iteration, rather than to recompute it entirely as in the standard Newton method or leave it unchanged as in the modified Newton method. Here we consider the update, known as the Broyden–Fletcher–Goldfarb–Shanno (BFGS). It is most conveniently written in terms of $\boldsymbol{K}^{(i+1)}$ rather than $\boldsymbol{K}^{(i)}$ and has the form

$$[\boldsymbol{K}^{(i)}]^{-1} = [\boldsymbol{I}+\boldsymbol{w}^{(i)}\{\boldsymbol{v}^{(i)}\}^T][\boldsymbol{K}^{(i-1)}]^{-1}[\boldsymbol{I}+\boldsymbol{v}^{(i)}\{\boldsymbol{w}^{(i)}\}^T]. \tag{12.23}$$

The indicated matrix multiplications are never carried out in the computer implementation; instead $\boldsymbol{v}^{(i)}$ and $\boldsymbol{w}^{(i)}$ are stored and used only in computing the new search direction

$$\Delta\boldsymbol{d}^{(i)} = [\boldsymbol{K}^{(i)}]^{-1}\boldsymbol{\psi}(\boldsymbol{d}^{(i)}). \tag{12.24}$$

A line search of the form given in (12.21) is adopted. The BFGS formulae for $\boldsymbol{v}^{(i)}$ and $\boldsymbol{w}^{(i)}$ are

$$\boldsymbol{v}^{(i)} = \boldsymbol{\psi}(\boldsymbol{d}^{(i)})\left(1+\alpha^{(i-1)}\left[\frac{\{\Delta \boldsymbol{d}^{(i-1)}\}^T \boldsymbol{\gamma}^{(i)}}{\{\boldsymbol{\delta}^{(i)}\}^T\{\boldsymbol{\psi}(\boldsymbol{d}^{(i-1)})\}}\right]^{1/2}\right) - \boldsymbol{\psi}(\boldsymbol{d}^{(i)}), \quad (12.25)$$

and

$$\boldsymbol{w}^{(i)} = \frac{\boldsymbol{\delta}^{(i)}}{\{\boldsymbol{\delta}^{(i)}\}^T \boldsymbol{\gamma}^{(i)}}, \quad (12.26)$$

where

$$\boldsymbol{\delta}^{(i)} = \boldsymbol{d}^{(i)} - \boldsymbol{d}^{(i-1)} = \alpha^{(i-1)} \Delta \boldsymbol{d}^{(i-1)},$$

and

$$\boldsymbol{\gamma}^{(i)} = \boldsymbol{\psi}(\boldsymbol{d}^{(i)}) - \boldsymbol{\psi}(\boldsymbol{d}^{(i-1)}).$$

The method has been successfully implemented and used by Matthies and Strang[91] and Geradin and Hogge[92] for both static and transient dynamic nonlinear problems. The stability of BFGS with respect to unloading has been emphasised by Matthies and Strang.[91] A related method by Crisfield[93] also shows much promise.

Rather than work with the inverse of $\boldsymbol{K}^{(i)}$ as given in (12.23), Geradin and Hogge[92] work with the update formula

$$\boldsymbol{K}^{(i)} = \boldsymbol{K}^{(i-1)} + \frac{\boldsymbol{\gamma}^{(i)}\{\boldsymbol{\gamma}^{(i)}\}^T}{\{\boldsymbol{\gamma}^{(i)}\}^T \boldsymbol{\delta}^{(i)}} - \frac{\{\boldsymbol{K}^{(i-1)}\boldsymbol{\delta}^{(i)}\}\{\boldsymbol{K}^{(i-1)}\boldsymbol{\delta}^{(i)}\}^T}{\{\boldsymbol{\delta}^{(i)}\}^T \boldsymbol{K}^{(i-1)} \boldsymbol{\delta}^{(i)}}, \quad (12.27)$$

and use a frontal solution scheme.

12.5.3 Conjugate gradient methods

In the conjugate gradient[94] algorithm we take

$$\boldsymbol{d}^{(i+1)} = \boldsymbol{d}^{(i)} + \alpha^{(i)}\boldsymbol{\delta}^{(i)}, \quad (12.28)$$

where

$$\boldsymbol{\delta}^{(i)} = \boldsymbol{\psi}(\boldsymbol{d}^{(i)}) + \beta^{(i)}\boldsymbol{\delta}^{(i-1)}, \quad (12.29)$$

in which $\alpha^{(i)}$ is chosen using a line search with the criterion that the total potential energy $\pi(\boldsymbol{d}^{(i+1)})$ should be minimised.

Initially, $\beta^{(0)}$ is set to zero. We list two possible values for $\beta^{(i)}$:

(i) The Hestenes–Stiefel[94] (Fletcher–Reeves[95]) algorithm

$$\beta^{(i)} = \frac{\{\boldsymbol{\psi}^{(i)}\}^T \boldsymbol{\psi}^{(i)}}{\{\boldsymbol{\psi}^{(i-1)}\}^T \boldsymbol{\psi}^{(i-1)}}. \quad (12.30)$$

(ii) The Polak–Ribiere[96] algorithm

$$\beta^{(i)} = \frac{\{\boldsymbol{\psi}^{(i)}\}^T \boldsymbol{\gamma}^{(i)}}{\{\boldsymbol{\psi}^{(i-1)}\}^T \boldsymbol{\psi}^{(i-1)}}. \quad (12.31)$$

The method, which requires modest computer core requirements, has been improved by scaling and other techniques.[97–99] The Conjugate–Newton method of Irons[100] is also a development of the basic conjugate gradient algorithm.

12.5.4 Other useful solution techniques

Among the remaining solution procedures, dynamic relaxation (DR) methods are quite popular. The main idea in DR originated from the observation that with about 90% of critical damping, an equivalent transient dynamic analysis rapidly converges to the steady state, static solution. Recent modifications[101–103] of the method have concentrated on finding improved replacements for the mass matrix M and the damping matrix C which are used in DR. Although DR methods are generally not as powerful as the various Newton and conjugate gradient methods, they require very little computer core storage and explicit transient dynamic programs such as DYNPAK, described in Chapter 10, can be rapidly modified to be used as DR solvers for *ad hoc* static problems when no other static program is available and results are urgently required.

It is usually difficult to decide on the form of load incrementation to adopt for elasto-plastic problems and exploratory analyses are often required. The work of Bergan and Soreide[104] in this area appears to be quite promising.

Schemes which work with local and global modes, several meshes or hierarchical representations[105–111] for the displacements may also prove to be of prime importance in nonlinear finite element equation solving.

12.6 Other enhancements in elasto-plastic analysis

12.6.1 Substructuring and boundary element methods

Economies can be made in the numerical solution of elasto-plastic problems by the use of substructuring techniques. A substructure analysis generally comprises the following steps.[112]

- Separate groups of elements within the solid are collectively identified as substructures as indicated in Fig. 12.10.
- For each substructure, the element stiffness matrices are assembled to give the global stiffness matrix of the substructure.
- The equations relating to the internal nodal points (i.e. nodes not on the boundary) are eliminated. This process is known as *condensation*.
- Solution of the system of resulting simultaneous equations is obtained by assembling all the individual substructures and any remaining elements which have not been associated with a substructure. This gives the nodal displacements and reactions for all nodal points on interfaces between substructures and for nodes of elements which are not related to any substructure.

- Return to the individual substructures to evaluate the displacements at interior nodes and finally obtain the element stresses.

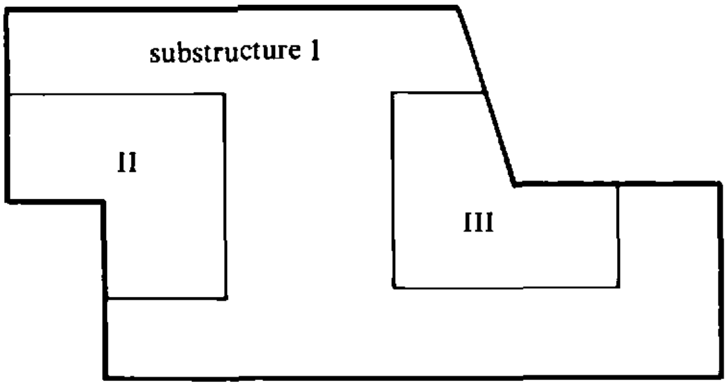


Fig. 12.10 Substructure analysis of elasto-plastic problems.

The very nature of the frontal equation solution process described in Section 6.4.12 makes the use of substructure techniques a simple affair, since, when the front has advanced into a structure to a certain position, the reduced frontal equations are essentially the condensed equations for a substructure corresponding to the part of the structure already considered.

For elasto-plastic problems, the part of the structure which (by physical considerations or experience!) is known to remain elastic during the deformation process can be defined as one substructure and the remaining elements considered individually. Thus during incremental/iterative solution the substructure stiffness will remain unaltered, for solution by the tangential stiffness method, and the substructure assembly and condensation process described above need be performed only once with an equation resolution process, necessitating only reduction of the R.H.S. terms being followed thereafter. The individual elements not associated with the substructure (and which model the elasto-plastic behaviour) are treated in the normal way as described in Chapter 7.

This approach can result in considerable computational economies, particularly if the mesh subdivision within the substructure is a fine one. It can be argued that a fine mesh subdivision is not warranted for regions where elastic behaviour is anticipated, but for structures which are to be subjected to more than one type of loading such an optimal mesh grading may not be possible. For example, with reference to Fig. 12.10, two separate loadings may cause plastic yielding in substructures II and III respectively and consequently a fine mesh grading within each of these regions cannot be avoided.

An extension of the above process is afforded by the use of the *boundary integral method.*[113–115] The boundary integral procedure requires trial functions which satisfy the governing equations directly and then attempt to satisfy the boundary conditions by a collocation, least-squares or Galerkin procedure. In order to find trial functions which satisfy the governing equations we are, at present, generally confined to linear elastic situations. Thus for the solution of elasto-plastic problems a coupled approach can be employed[113,115] with the elastic region of the structure being modelled by boundary elements and conventional finite elements employed to treat the elasto-plastic zones. Such direct coupling leads to nonsymmetric matrices which is acceptable if the equation set is dominated by the boundary integral equations.

This approach promises efficient numerical solutions particularly for cases of limited yielding in three-dimensional solids where the surface area/volume ratio is relatively small. The process can also be used to advantage in infinite domain structures such as rock mass problems or soil/structure interaction problems with boundary elements being employed to model the exterior domain.

12.6.2 Interactive computing

The solution of elasto-plastic problems inevitably requires some degree of insight into the structural behaviour before choice of solution parameters, such as load increment sizes, can be made. Even then it is difficult, if not impossible, to specify the most suitable values of load increments, tolerance factors for each load case and also choice of the optimal solution process (e.g. initial stiffness, tangential stiffness or some combined algorithm) is equally difficult to arrive at.

To this end, the developments which are currently taking place in interactive computing will become increasingly important. Here we envisage the situation where the results for a particular load increment are held in core while the solution is scrutinized. Depending on the convergence characteristics, etc., the load increment size and convergence tolerance factor are then input and solution continued for a further increment. If required the nonlinear solution process can be redefined at this stage changing, for example, from a tangential stiffness to an initial stiffness algorithm if collapse conditions are being approached. Furthermore if the numerical process did not converge in the previous increment, the calculations could be repeated for a smaller load increment size or a different solution algorithm.

12.6.3 Computational techniques

Many new and improved programming strategies are developing in connection with finite element software and the interested reader is directed to the work of Schrem[116,117] and others[118] who are active in this area.

12.7 Concluding remarks

Throughout this text we have described numerical techniques and computer codes for a variety of engineering applications. Treatment has been limited to situations where the finite element method can be used to provide nonlinear solutions with a measure of confidence. In this final chapter we have attempted to indicate some areas of further study and here the applicability to design problems is not so clear. For example, for soils and concrete some divergence of opinion still exists as to selection of an appropriate material model. Indeed at the present time it is true to say that numerical solution capabilities are in advance of the knowledge of fundamental material behaviour. This is particularly true for dynamic problems where there is a scarcity of information on material response under transient conditions. In this respect it would appear that nonlinear finite element methods offer the possibility of conducting 'numerical experiments' to provide insight on material behaviour which could not be obtained by experiment alone.

12.8 References

1. NAYLOR, D. J., Stress–strain laws for soil, In: *Developments in Soil Mechanics*, Ed. C. R. Scott, Applied Science Publishers (1978).
2. SCHOFIELD, A. N. and WROTH, C. P., *Critical State Soil Mechanics*, McGraw-Hill, New York (1968).
3. ATKINSON, J. H. and BRANSBY, P., *Mechanics of Soil*, McGraw-Hill, New York (1979).
4. NILSSON, L., Impact loading on concrete structures, Publication 79: 1, Dept. of Structural Mechanics, Chalmers University of Technology, Sweden (1979).
5. ZIENKIEWICZ, O. C., OWEN, D. R. J., PHILLIPS, D. V. and NAYAK, G. C., Finite element methods in the analysis of reactor vessels, *Nuclear Engng. and Design*, **20**, 507–542 (1972).
6. MOHRAZ, B., SCHNOBRICH, W. C. and GOMEZ, A. E., Crack development in a prestressed concrete reactor vessel as determined by a lumped parameter method, *Nuclear Engng. and Design*, **11**, 286 (1970).
7. VALANIS, K. C., A theory of viscoplasticity without a yield surface, *Archiwum Mechaniki Stossowanej*, **23**, 4, 517–533 (Warsaw, 1971).
8. VALANIS, K. C., On the foundations of the endochronic theory of viscoplasticity, *Archiwum Mechaniki Stossowanej*, **27**, 5–6, 857–869 (Warsaw, 1975).
9. BAZANT, Z. P., A new approach to inelasticity and failure of concrete, sand and rock: Endochronic theory, *Proc. Soc. of Engng. Science, Eleventh Annual Meeting*, Ed. G. J. Dvorak, Duke University, Durham, North Carolina (1974).
10. BAZANT, Z. P., BHAT, P. D. and SHIEH, C. L., Endochronic theory for inelasticity and failure analysis of concrete structures, Struct. Engng. Report No. 1976-12/259, Department of Civil Engineering, Northwestern University (1976).
11. ZIENKIEWICZ, O. C., *The Finite Element Method*, McGraw-Hill, London (1977).
12. ZIENKIEWICZ, O. C., Viscoplasticity, plasticity, creep and viscoplastic flow, *Proc. Int. Conf. on Computational Methods in Nonlinear Mechanics*, University of Texas, Austin (1974).
13. PRICE, J. W. H. and ALEXANDER, J. M., The finite element analysis of two high temperature metal deformation processes, *Proc. Second Int. Symp. on Finite Element Methods in Flow Problems*, Santa Margharita Ligure, Rappelo, Italy (June 1976).
14. HINTON, E., SALONEN, E. M. and BICANIC, N., A study of locking phenomena in isoparametric elements, In: *The Mathematics of Finite Elements and Applications III*, 437–447, Ed. J. R. Whiteman, Academic Press, London (1979).

15. SKELLAND, A. N. P., *Non-Newtonian Flow and Heat Transfer*, John Wiley, New York (1967).
16. BIRD, R. B., STEWART, W. E. and LIGHTFOOT, E. N., *Transport Phenomena*, John Wiley, New York (1960).
17. LYNESS, J. F., OWEN, D. R. J. and ZIENKIEWICZ, O. C., The finite element analysis of engineering systems governed by a nonlinear quasi-harmonic equation, *Computers and Structures*, **5**, 65–79 (1975).
18. LAIRD, W. M., Slurry and suspension transport, *Ind. Engng. Chem.* **49**, 1, 138–141 (1957).
19. *Journal of Strain Analysis* (*Special Issue*), Vol. 10, No. 4 (October, 1975).
20. KNOTT, J. F., *Fundamentals of Fracture Mechanics*, Butterworths, London (1973).
21. RICE, J. R., A path independent integral and the approximate analysis of strain concentration by notches and cracks, *J. Appl. Mechanics*, 379–386 (1968).
22. HUTCHINSON, J. W., Singular behaviour at the end of a tensile crack in a hardening material, *J. Mech. Phys. Solids*, **16**, 13–31 (1968).
23. RICE, J. R. and ROSENGRUN, G. F., Plane strain deformation near a crack tip in a power-law hardening material, *J. Mech. Phys. Solids*, **16**, 1–12 (1968).
24. MCCLINTOCK, F., *Fracture, An Advanced Treatise*, Chapter 2, Ed. H. Liebowitz, Academic Press, New York (1971).
25. HILL, R., *The Mathematical Theory of Plasticity*, Oxford University Press (1950).
26. LEVY, N., MARCAL, P. V., OSTERGREN, W. J. and RICE, J. R., Small scale yielding near a crack in plane strain: a finite element analysis, *Int. J. Fract. Mech.* **9**, 98–100 (1973).
27. RICE, J. R. and TRACEY, D. M., Computational fracture mechanics, In: *Numerical and Computer Methods in Structural Mechanics*, Eds. S. J. Fenves *et al.*, Academic Press (1973).
28. ZIENKIEWICZ, O. C., Finite elements in the time domain, Contribution for 'State-of-Art' Survey by the Committee in Computing in Applied Mechanics of ASME (to be published).
29. PARK, K. C., Partitional transient analysis procedures for coupled field problems, Report No. LMSC-D633955, Lockheed Palo Alto Laboratory (1979).
30. PARK, K. C., FELLIPA, C. A. and DERUNTZ, J. A., Stabilization of staggered solution procedures for fluid-structure interaction analysis, In: *Computer Methods for Fluid-Structure Interaction Problems—AMD*, Vol. 26, Eds. T. Belytschko and T. L. Geers (1977).
31. HUGHES, T. J. R., Recent developments in computer methods for structural analysis, Division M Principal Lecture, Fifth SMIRT Conference, Berlin (to appear in *Nuclear Engng. and Design*).
32. ZIENKIEWICZ, O. C., CHANG, C. T., HINTON, E. and LEUNG, K. H., Effective stress dynamic modelling for soil structures including drainage and liquefaction, In: *Proc. of Int. Symp. on Soils Under Cyclic and Transient Loading*, Eds. G. N. Pande and O. C. Zienkiewicz, 551–554, A. A. Balkema, Rotterdam (1980).
33. FREDRIKSSON, B. and MACKERLE, J., *Finite Element Review*, Publication No. AEC-L-003, AEC, Box 3944, S-58903, Linköping, Sweden.
34. ASHWELL, D. G. and GALLAGHER, R. H. (Eds.), *Finite Elements for Thin Shells and Curved Members*, Wiley, London (1976).
35. ARMEN, H., PIFKO, A., and LEVINE, H. S., A finite element method for the plastic bending analysis of structures, *Proc. Second Conf. on Matrix Methods*

in Structural Mechanics, DOC.AFFDL-TR-69-150, Wright-Patterson Air Force Base, Ohio, 1969.

36. BUSHNELL, D., Buckling of elastic-plastic complex shells of revolution including large deflections and creep, *Computers and Structures*, **6**, 221–239 (1976).
37. WEGMULLER, A. W. and KOSTEM, C. N., Finite element analysis of elasto-plastic plates and eccentrically stiffened plates, Report No. 378A.4, Fritz Engg. Laboratory, Lehigh University (1973).
38. BERGAN, P. G., Application of finite element method to nonlinear problems, *Finite Element Congress*, Baden-Baden, FRG, 18–19 Nov. (1974).
39. CRISFIELD, M. A., Large deflection elasto-plastic buckling analysis of plates using finite elements, TRRL Report LR 593, Crowthorne (1973).
40. CRISFIELD, M. A., Some approximations in the nonlinear analysis of rectangular plates using finite elements, TRRL Suppl. Report 51 UC, Crowthorne (1974).
41. CRISFIELD, M. A., On an approximate yield criterion for thin steel shells, TRRL Report 658, Crowthorne (1974).
42. CRISFIELD, M. A., Large deflection elasto-plastic buckling analysis of eccentrically stiffened plates using finite elements, TRRL Report 725, Crowthorne (1976).
43. CRISFIELD, M. A. and PUTALI, R. S., Approximations in the nonlinear analysis of thin plated structures, *Int. Conf. Finite Elements in Nonlinear Solid and Structural Mechanics*, Geilo, Norway (1977).
44. CRISFIELD, M. A., Ivanov's yield criterion for thin plates and shells using finite elements, TRRL Lab. Report 919, Crowthorne (1979).
45. BERGAN, P. G., Nonlinear analysis of plates considering geometric and material effects, Ph.D. Thesis, University of California, Berkeley, SESM Report No. 71-7 (1971).
46. HORRIGMOE, G. and BERGAN, P. G., Incremental variational principles and finite element models for nonlinear problems, *Comp. Meths. Appl. Mech. and Engng.* **7**, 201–217 (1976).
47. HORRIGMOE, G., Large displacement analysis of shells by a hybrid stress finite element method, *IASS World Congress on Space Enclosures* (*NCOSE-76*, Montreal, 489–499 (1976).
48. HORRIGMOE, G., Finite element instability analysis of free-form shells, Report No. 77-2, The Norwegian Inst. Techn., University of Trondheim (1977).
49. WOOD, R. D., The application of finite element methods to geometrically nonlinear structural analysis, Ph.D. Thesis, University College of Swansea (1973).
50. PICA, A., Geometrically nonlinear analysis of Mindlin plates by finite element method, M.Sc. Thesis, University College of Swansea (1978).
51. PICA, A., WOOD, R. D. and HINTON, E., Finite element analysis of geometrically nonlinear plate behaviour using a Mindlin formulation, *Computers and Structures*, **11**, 203–215 (1980).
52. GALLAGHER, R. H., Geometrically nonlinear shell analysis, In: *Advances in Computational Methods in Structural Mechanics and Design*, Eds. J. T. Oden, R. W. Clough and Y. Yamamoto, Univ. Alabama Press, 641–678 (Proc. Second US–Japan Seminar) (1972).
53. FREY, F. and CESCOTTO, S., Some new aspects of the incremental total Lagrangian description in nonlinear analysis, *Proc. Int. Conf. Finite Elements in Nonlinear Solid Mechanics*, Geilo, Norway, Paper CO5 (1977).
54. BATOZ, J. L., CATTOPADHYAY, A. and DHATT, G., Finite element large deflection analysis of shallow shells, *Int. J. Num. Meth. Engng.* **10**, 39–58 (1976).

55. ARGYRIS, J. H. and DUNNE, P. C., Non-linear and post-buckling analysis of structures, In: *Formulations and Computational Algorithms in Finite Element Analysis*, Eds. K. J. Bathe, J. T. Oden and W. Wunderlich, 525–571, MIT Press, Massachusetts (1977).
56. AHMAD, S., IRONS, B. M. and ZIENKIEWICZ, O. C., Curved thick shell and membrane elements with particular reference to axisymmetric problems, *Proc. Second Conf. Matrix Method Struct. Mech.*, Wright-Patterson Air Force Base, Ohio (1968).
57. ZIENKIEWICZ, O. C., TAYLOR, R. L. and TOO, J. M., Reduced integration technique in general analysis of plates and shells, *Int. J. Num. Meth. Engng.* **3**, 275–290 (1971).
58. HINTON, E., RAZZAQUE, A., ZIENKIEWICZ, O. C. and DAVIES, J. D., A simple finite element solution for plates of homogeneous, sandwich and cellular construction, *Proc. Inst. Civ. Engrs. Part* 2, **59**, 43–65 (1975).
59. ZIENKIEWICZ, O. C. and HINTON, E., Reduced integration, function smoothing and non-conformity in finite element analysis (with special reference to thick plates), *J. of the Franklin Inst.*, **302**, 443–461 (1976).
60. HINTON, E., OWEN, D. R. J. and SHANTARAM, D., Dynamic transient nonlinear behaviour of thick and thin plates, In: *The Mathematics of Finite Elements and Applications II MAFELAP*, Ed. J. R. Whiteman, 423–438, Academic Press, London (1977).
61. HINTON, E. and PUGH, E. D. L., Some quadrilateral isoparametric finite elements based on Mindlin plate theory, *Proc. of Symp. on Applications of Computer Methods in Engineering*, Univ. of S. California, Vol. II, 851–858 (1977).
62. BICANIC, N. and HINTON, E., Spurious modes in two-dimensional isoparametric elements, *Int. J. Num. Meths. Engng.* **14**, 1545–1557 (1979).
63. HINTON, E. and BICANIC, N., A comparison of Lagrangian and Serendipity Mindlin plate elements for free vibration analysis, *Computers and Structures*, **10**, 483–493 (1979).
64. HUGHES, T. J. R., COHEN, M. and HAROUN, M., Reduced and selective integration techniques in the finite element analysis of plates, *Nuclear Engng. Design*, **46**, 203–222 (1978).
65. HUGHES, T. J. R. and COHEN, M., The 'Heterosis' finite element for plate bending, *Computers and Structures*, **9**, 445–450 (1978).
66. MALKUS, D. and HUGHES, T. J. R., Mixed finite element methods—reduced selective integration techniques: a unification of concept, *Comp. Appl. Mech. Engng.* **15**, 63–81 (1978).
67. RAMM, E., A plate/shell element for large deflection and rotations, In: *Formulations and Computational Algorithms in Finite Element Analysis*, Eds. K. J. Bathe, J. T. Oden and W. Wunderlich, 264–293, MIT Press, Massachusetts (1977).
68. KRAKELAND, B., Large displacement analysis of shells considering elasto-plastic and elasto-viscoplastic materials, Report No. 776, The Norwegian Institute of Technology, The University of Trondheim, Norway (1977).
69. BATHE, K. J. and BOLOURCHI, S., A geometric and material nonlinear plate and shell element (to be published).
70. LARSEN, P. K. and POPOV, E. P., Large displacement analysis of viscoelastic shells of revolution, *Comp. Meth. Appl. Mech. Engng.* **3**, 237–253 (1974).
71. NOOR, A. K. and HARLEY, S. J., Nonlinear shell analysis via mixed isoparametric elements, *Computers and Structures*, **7**, 615–626 (1977).

72. FREY, F., L'analyse statique non linéaire des structures par la méthode des éléments finis et son application à la construction metallique, Ph.D. Thesis, University of Liège, Belgium (1978).
73. BOLOURCHI, S., On finite element nonlinear analysis of general shell structures, Ph.D. Thesis, Mechanical Engineering Department, MIT (1979).
74. MARTINS, R. A. F. and OWEN, D. R. J., Elastoplastic and geometrically nonlinear thin shell analysis by the Semiloof element, *Computers and Structures* (to be published).
75. DINIS, L. M. S., MARTINS, R. A. F. and OWEN, D. R. J., Analysis of material and geometrically nonlinear thin plates and arbitrary shells, *Proc. Int. Conf. Numerical Methods for Nonlinear Problems*, Swansea, 425–442, Pineridge Press, Swansea, U.K. (1980).
76. OWEN, D. R. J. and DINIS, L. M. S., Transient dynamic analysis of thin shells including viscoplastic and large displacement effect, In: *The Mathematics of Finite Elements and Applications III*, Ed. J. R. Whiteman, Academic Press (1978).
77. WEGNER, R., Finite element models for reinforced concrete, In: *Formulation and Computational Algorithms in Finite Element Analysis*, Eds. K. J. Bathe, J. T. Oden and W. Wunderlich, 393–439, MIT Press, Massachusetts (1977).
78. JOFREIT, J. C. and MCNIECE, G. M., Finite element analysis of reinforced concrete slabs, *J. Struct. Div. ASCE*, **97**, 785–806 (1971).
79. HAND, F. R., PENCOLD, D. A. and SCHONBRICH, W. C., Non-linear layered analysis of reinforced concrete plates and shells, Univ. Illinois, Struct. Research Report No. 389 (1972).
80. LIN, C. S., Nonlinear analysis of reinforced concrete slabs and shells, Report UC SESM 73-7, Univ. of California, Berkeley (1973).
81. COPE, R. J. and RAO, P. U., Non-linear finite element analysis of concrete slab structures, *Proc. Inst. Civ. Engrs. Part* 2, **63**, 149–179 (1977).
82. SCHNOBRICH, W. C., Behaviour of reinforced concrete structures predicted by the finite element method, *Computers and Structures*, **7**, 365–376 (1977).
83. BASHUR, F. K. and DARWIN, D., Non-linear model for reinforced concrete slabs, *J. Struct. Divn., ASCE*, **104**, 157–170 (1978).
84. GILBERT, R. F. and WARNER, R. F., Tension stiffening in reinforced concrete slabs, *J. Struct. Divn. ASCE*, **104**, 1885–1900 (1978).,
85. DUNCAN, W. and JOHNARRY, T., Further studies on the constant stiffness method of non-linear analysis of concrete structures, *Proc. Inst. Civ. Engrs.. Part* 2, **67**, 951–969 (1979).
86. CORMEAU, I. C., Viscoplasticity and plasticity in the finite element method, Ph.D. Thesis, University College of Swansea (1976).
87. DINIS, L. M. S., Finite element viscoplastic analyses of plates, M.Sc. Thesis, University College of Swansea (1975).
88. NAYAK, G. C., Plasticity and large deformation problems by finite element method, Ph.D. Thesis, University of Wales, Swansea (1971).
89. BASU, A. K., Letter to the Editor, 'New light on the Nayak alpha technique', *Int. J. Num. Meth. Engng.* **6**, 152 (1973).
90. IRONS, B. M. and AHMAD, S., *Techniques of Finite Elements*, Ellis Horwood, Chichester (1980).
91. MATTHIES, H. and STRANG, G., The solution of nonlinear finite element equations, *Int. J. Num. Meth. Engng.* **14**, 1613–1626 (1979).
92. GERADIN, M. and HOGGE, M. A., Quasi-Newton iteration in non-linear structural dynamics, Paper M7/1, *Trans. Fifth Int. Conf. on SMIRT*, Berlin, North-Holland, Amsterdam (1979).

93. Crisfield, M. A., Iterative solution procedure for linear and non-linear structural analysis, TRRL Lab. Report 900, Transport and Road Research Laboratory, Crowthorne, Berkshire, U.K. (1979).
94. Hestenes, M. and Stiefel, E., Method of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards*, **49**, 409–436 (1952).
95. Fletcher, R. and Reeves, C. M., Function minimisation by conjugate gradients, *The Computer Journal*, **7**, 149–154 (1964).
96. Polak, E., *Computational methods in optimisation: a unified approach*, Academic Press, London (1971).
97. Jennings, A. and Malik, G. M., The solution of sparse linear equations by the conjugate gradient method, *Int. J. Num. Meth. Engng*. **12**, 141–158 (1978).
98. Fried, I. and Metzler, J., SOR v. conjugate gradients in a finite element discretization, *Int. J. Num. Meth. Engng*. **12**, 1329–1342 (1978).
99. Kershaw, D. S., The incomplete Cholesky–conjugate gradient method for the iterative solution of systems of linear equations, *J. Comp. Phys*., **26**, 43–65 (1978).
100. Irons, B. M. and Elsawaf, A. F., The conjugate Newton algorithm for solving finite element equations, In: *Formulations and Computational Algorithms in Finite Element Analysis*, Eds. K. J. Bathe, J. T. Oden and W. Wunderlich, MIT Press, 655–672 (1977).
101. Brew, J. S. and Brotton, D. M., Non-linear structural analysis by dynamic relaxation, *Int. J. Num. Meth. Engng*. **3**, 463–483 (1971).
102. Pica, A. and Hinton, E., Transient and pseudo-transient analyses of Mindlin plates: Theory and applications, *Int. J. Num. Meth. Engng*. **15**, 189–208 (1980).
103. Underwood, P., An adaptive dynamic relaxation technique for nonlinear structural analysis, Draft Report, Lockheed Palo Alto Research Laboratory.
104. Bergan, P. G. and Soreide, T., Solution of large displacement and instability problems using current stiffness parameter, *Proc. Conf. on Finite Elements in Nonlinear Solid and Structural Mechanics*, Geilo, Norway (1977).
105. Mote, C. D. J., Global–local finite element, *Int. J. Num. Meth. Engng*. **3**, 565–574 (1971).
106. Crisfield, M. A., A combined Rayleigh–Ritz/finite element method for the non-linear analysis of stiffened plate structures, *Computers and Structures*, **8**, 679–689 (1978).
107. Wilson, E. L., Special numerical and computer techniques for the analysis of finite element systems, *Proc. US–German Symposium on Formulations and Algorithms in Finite Element Analysis*, Eds. K. J. Bathe, T. J. Oden and W. Wunderlich, 3–25, MIT Press, Boston (1977).
108. Wachspress, E. L., Two-level finite element computation, *Proc. US–German Symposium on Formulations and Algorithms in Finite Element Analysis*, Eds. K. J. Bathe, T. J. Oden and W. Wunderlich, 877–913, MIT Press, Boston (1977).
109. Brandt, A., Multi-level adaptive solutions to boundary-value problems, *Math. Comptr*., **31**, 333–390 (1977).
110. Felippa, C. A., Procedures for computer analysis of large nonlinear structural analysis, *Int. Symp. on Large Engineering Systems*, The University of Manitoba, Winnipeg, Canada, 1976, Pergamon Press, Oxford (1977).
111. Peano, A. M., Fanelli, M., Riccioni, R. and Sandella, L., Self-adaptive convergence at the crack tip of a dam buttress, *Proc. of First Int. Conf. on Numerical Methods in Fracture Mechanics*. Eds. A. R. Luxmoore and D. R. J. Owen, 268–280, Pineridge Press, Swansea, U.K. (1978).

112. Row, D. G. and Powell, G. H., A substructure technique for nonlinear static and dynamic analysis, Report No. UCB/EER/C-78/15, University of California, Berkeley (1978).
113. Zienkiewicz, O. C., Kelly, D. W. and Bettess, P., 'Marriage à la mode', Finite elements and boundary integrals, *Proc. Conf. Innovative Numerical Analysis in Engineering Science*, CETIM, Paris (1977).
114. Banerjee, P. K., Cathie, D. N. and Davies, T. G., Two- and three-dimensional problems of elastoplasticity, In: *Developments in Boundary Integral Methods*, Chapter 4, Eds. P. K. Banerjee and R. Butterfield, Applied Science Publishers (1979).
115. Kelly, D.W., Mustoe, G. G. W. and Zienkiewicz, O. C., Coupling boundary element methods with other numerical methods, In: *Developments in Boundary Integral Methods*, Chapter 10, Eds. P. K. Banerjee and R. Butterfield, Applied Science Publishers (1979).
116. Schrem, E., Trends and aspects of the development of large finite element software systems, *Computers and Structures*, **10**, 419–425 (1979).
117. Schrem, E., From program systems to programming systems for finite element analysis, In: *Formulations and Computational Algorithms in Finite Element Analysis*, Eds. K. J. Bathe, J. T. Oden and W. Wunderlich, MIT Press, 163–189 (1977).
118. Zave, P. and Rheinboldt, W. C., Design of an adaptive, parallel finite-element system, *ACM Transactions on Mathematical Software*, **5**, No. 1, 1–17 (1979).

Appendix I

Instructions for preparing input data for one-dimensional problems

In Part I of this text computer codes have been presented for the nonlinear analysis of several classes of one-dimensional problems. In Chapter 3 the data structure for the following applications was discussed:

- Direct iteration solution of nonlinear quasiharmonic problems.
- Use of the Newton–Raphson process for the solution of nonlinear quasiharmonic problems.
- Nonlinear elastic applications.
- Elasto-plastic material behaviour.

In Chapter 4 the time transient phenomenon of one-dimensional visco-plasticity was discussed. In Chapter 5 solution techniques were presented for elasto-plastic beam bending problems. In this appendix user instructions for preparing input data for each of these applications are provided.

A.1.1 Program QUITER for the solution of nonlinear one-dimensional quasiharmonic problems by direct iteration

CARD SET 1 TITLE CARD (12A6)—One card

Cols. 1–72 Title of the problem—limited to 72 alphanumeric characters.

CARD SET 2 CONTROL CARD (9I5)—One card

Cols. 1–5	NPOIN	Total number of nodal points.
6–10	NELEM	Total number of elements.
11–15	NBOUN	Total number of restrained boundary points—nodes at which the value of the unknown (e.g. temperature) is prescribed.
16–20	NMATS	Total number of different materials.
21–25	NPROP	Number of independent properties per material (= 1).
26–30	NNODE	Number of nodes per element (= 2).
31–35	NINCS	Number of increments in which the total 'loading' is to be applied.

36–40	NALGO	Nonlinear solution process indicator (= 1, for solution by direct iteration).
41–45	NDOFN	Number of degrees of freedom per node (= 1).

CARD SET 3 MATERIAL CARDS (I5, F15.5)—One card for each different material. Total of NMATS cards (See Card Set 2).

Cols. 1–5	JMATS	Material identification number.
6–20	PROPS(JMATS,1)	The material coefficient, K_0 in (2.27).

CARD SET 4 ELEMENT CARDS (4I5)—One card for each element. Total of NELEM cards (See Card Set 2).

Cols. 1–5	JELEM	Element number.
6–10	LNODS(JELEM,1)	1st nodal connection number.
11–15	LNODS(JELEM,2)	2nd nodal connection number.
16–20	MATNO(JELEM)	Material property number.

NOTE: The two nodal connection numbers for an element can be taken in any order.

CARD SET 5 NODAL COORDINATE CARDS (I10,F15.5)—One card for each node. Total of NPOIN cards (See Card Set 2).

Cols. 1–10	JPOIN	Node number.
11–25	COORD(JPOIN)	The x coordinate of the node.

Note: The origin of the coordinate system may be arbitrarily located.

CARD SET 6 RESTRAINED NODE CARDS (I10,I5,F10.5)—One card for each restrained node. Total of NBOUN cards (See Card Set 2).

Cols. 1–10	NODFX	Restrained node number.
11–15	ICODE(1)	Condition of restraint(= 1).
16–25	PRESC(1)	The prescribed value of the nodal variable.

CARD SET 7 APPLIED 'LOAD' CARDS (I10,2F15.5)—One card for each loaded element.

Cols. 1–10	IELEM	The element number.
11–25	RLOAD(IELEM,1)	The applied load at the 1st node of the element.
26–40	RLOAD(IELEM,2)	The applied load at the 2nd node of the element.

Notes: 1) The 1st and 2nd nodes must be taken in the order listed in Card Set 4.

2) This card set must terminate with data for the highest numbered element whether it is loaded or not.

CARD SET 8 LOAD INCREMENT CONTROL CARDS (2I5,2F15.5)—One card for each load increment. Total of NINCS cards (See Card Set 2).

Cols. 1–5	NITER	Maximum number of iterations allowed for the 'load' increment.
6–10	NOUTP	Output control parameter: 1—Results output only after the first iteration and after convergence, 2—Results output after each iteration.
11–25	FACTO	Applied 'load' factor for the increment—specified as a factor of the loading input in Card Set 7.
26–40	TOLER	Convergence tolerance factor.—The term TOLER in (3.21).

Note: The applied loading factors are accumulative. If FACTO is specified as 0.6, 0.3, 0.3 for the first three 'load' increments, then the total loading acting during the third increment is 1.2 times that specified in Card Set 7.

If the form of the material nonlinearity is to be changed, then FUNCTION VARIA must be modified in accordance with the process described in Section 3.9.1.

A.1.2 Program QUNEWT for the solution of nonlinear one-dimensional quasiharmonic problems by the Newton–Raphson process

Data input for this application is identical to that described in Section A.1.1 above with the following exceptions:

CARD SET 2 CONTROL CARD

Cols. 21–25	NPROP	Number of independent properties per material (= 2).
36–40	NALGO	Nonlinear solution process parameter (= 2, for Newton–Raphson solution technique).

CARD SET 3 MATERIAL CARDS (I5,2F15.5)—One card for each different material.

Cols. 1–5	JMATS	Material identification number.
6–20	PROPS(JMATS,1)	The material coefficient K_0 in (2.27).
21–35	PROPS(JMATS,2)	The term b in (2.27).

A.1.3 Program NONLAS for the solution of one-dimensional nonlinear elastic problems

The input data for this application is again identical to that described in Section A.1.1 with the following exceptions. The basic nodal variable is now the axial displacement.

CARD SET 2 CONTROL CARD

Cols. 21–25	NPROP	Number of independent properties per material(= 2).
36–40	NALGO	Nonlinear solution process indicator:
		1 or 2 *Tangential stiffness algorithm.* The element stiffnesses are recalculated for each iteration of the solution process.
		3 *Initial stiffness method.* The stiffnesses are calculated at the beginning of the solution process and maintained constant thereafter.
		4 *Combined algorithm* (*Version I*). The element stiffnesses are recomputed for the *first* iteration of each load increment.
		5 *Combined algorithm* (*Version II*). The element stiffnesses are recomputed for the *second* iteration of each load increment.

CARD SET 3 MATERIAL CARDS (I5,2F15,5)—One card for each different material.

Cols. 1–10	JMATS	Material identification number.
6–20	PROPS(JMATS,1)	Elastic modulus, E.
21–35	PROPS(JMATS,2)	Cross-sectional area, A.

A.1.4 Program ELPLAS for the solution of one-dimensional elastoplastic problems

The input data for this application is again identical to that described in Section A.1.1 with the following exceptions. The basic nodal variable is the axial displacement.

CARD SET 2 CONTROL CARD (9I5)

Cols. 21–25	NPROP	Number of independent properties per material (= 4).
36–40	NALGO	Nonlinear solution process indicator:
		1 or 2 Tangential stiffness algorithm.

3	Initial stiffness method.
4	Combined algorithm with stiffnesses recomputed for the 1st iteration.
5	Combined algorithm with stiffnesses recomputed for the 2nd iteration.

CARD SET 3 MATERIAL CARDS (I5,4F15.5)—One card for each different material.

Cols. 1–5	JMATS	Material identification number.
6–20	PROPS(JMATS,1)	Elastic modulus, E.
21–35	PROPS(JMATS,2)	Cross-sectional area, A.
36–50	PROPS(JMATS,3)	Uniaxial yield stress, σ_Y.
51–65	PROPS(JMATS,4)	Linear strain-hardening parameter, H'.

A.1.5 Program UNVIS for the solution of one-dimensional elasto-viscoplastic problems

The input data for this application is once again identical to that described in Section A.1.1 with the following exceptions. The basic nodal variable is the axial displacement.

CARD SET 2 CONTROL CARD

Cols. 21–25	NPROP	Number of independent properties per material (= 5).
36–40	NALGO	Nonlinear solution process indicator (= 1, for Euler time stepping scheme).

CARD SET 3 MATERIAL CARDS (I5,5F15.5)—One card for each different material.

Cols. 1–5	JMATS	Material identification number.
6–20	PROPS(JMATS,1)	Elastic modulus, E.
21–35	PROPS(JMATS,2)	Cross-sectional area, A.
36–50	PROPS(JMATS,3)	Uniaxial yield stress, σ_Y.
51–65	PROPS(JMATS,4)	Linear strain-hardening parameter, H'.
66–80	PROPS(JMATS,5)	Fluidity parameter, γ.

CARD SET 8 TIMESTEPPING PARAMETER CARD (3F15.5)—One card.

Cols. 1–15	TAUFT	The factor τ employed to limit the time-step length according to (4.38).
16–30	DTINT	The initial time step length (required to initiate the time stepping process.
31–45	FTIME	The factor k in (4.39).

CARD SET 9 LOAD INCREMENT CONTROL CARDS

This card set is identical to Card Set 8, Section A.1.1 where the term 'iteration' is now replaced by 'timestep'.

A.1.6 Program TIMOSH for the nonlayered elasto-plastic analysis of Timoshenko beams

The input data for this application is identical to that described in Section A.1.1 with the following exceptions.

CARD SET 2 CONTROL CARD (9I5)

Cols. 21–25	NPROP	Number of independent properties per material (=4)
36–40	NALGO	Nonlinear solution process indicator: 1 or 2 Tangential stiffness algorithm. 3 Initial stiffness method. 4 Combined algorithm with stiffnesses recomputed for the 1st iteration. 5 Combined algorithm with stiffnesses recomputed for the 2nd iteration.
41–45	NDOFN	Number of degrees of freedom per node (=2).

CARD SET 3 MATERIAL CARDS (I5, 4F15.5)—One card for each different material.

Cols. 6–20	PROPS(JMATS, 1)	Flexural rigidity, EI.
21–35	PROPS(JMATS,2)	Shear constant, $GA/1.5$.
36–50	PROPS(JMATS, 3)	Yield moment, M_0.
51–65	PROPS(JMATS, 4)	Strain hardening parameter, H'.

CARD SET 6 RESTRAINED NODE CARDS (I10, 2(I5, F10.5))—One card for each restrained node. Total of NBOUN cards.

Cols. 11–15	ICODE(1)	Condition of restraint on nodal displacement, w. 0—No displacement restraint. 1—Nodal displacement restrained.
16–25	VALUE(1)	The prescribed value of nodal displacement, w.
26–30	ICODE(2)	Condition of restraint on nodal rotation, θ. 0—No rotation restraint. 1—Nodal rotation restrained.
31–40	VALUE(2)	The prescribed value of nodal rotation, θ.

CARD SET 7 APPLIED LOAD CARDS (I10, 4F15.5)—One card for each loaded element.

Cols.	1–10	JELEM	Element number.
	11–25	RLOAD(JELEM,1)	Transverse load applied at the first node.
	26–40	RLOAD(JELEM,2)	Couple applied at the first node.
	41–55	RLOAD(JELEM,3)	Transverse load applied at the second node.
	56–70	RLOAD(JELEM,4)	Couple applied at the second node.

Note: The last card should be that for the highest numbered element whether it is loaded or not.

A.1.7 Program TIMLAY for the layered elasto-plastic analysis of Timoshenko beams

The input data for this application is identical to that described in Section A.1.6 with the following exceptions.

CARD SET 2 CONTROL CARD (10I5)

Cols.	21–25	NPROP	Number of independent properties per material ($=4+2\times$Total number of layers).
	46–50	NLAYR	Total number of layers.

CARD SET 3 MATERIAL CARDS

1st Card (I5, 4F15.5)

Cols.	1–5	NUMAT	Material identification number.
	6–20	PROPS(NUMAT,1)	Young's modulus, E.
	21–35	PROPS(NUMAT,2)	Modified shear modulus, $G/1.5$.
	36–50	PROPS(NUMAT,3)	Yield stress, σ_Y.
	51–65	PROPS(NUMAT,4)	Strain hardening parameter, H'.

2nd and subsequent cards (4F15.5)

Cols.	1–15	BRDTH(1)	Breadth of the 1st layer.
	16–30	THICK(1)	Thickness of the 1st layer.
	31–45	BRDTH(2)	Breadth of the 2nd layer.
	.	.	
	.	.	
	.	.	
	.	BRDTH(NLAYR)	Breadth of the last layer.
	.	THICK(NLAYR)	Thickness of the last layer.

Appendix II

Instructions for preparing input data for plane, axisymmetric and plate bending problems

In this appendix user instructions are provided for the computer programs developed in Part II of this text. Chapter 7 dealt with elasto-plastic problems in two dimensions and in Chapter 8 the corresponding time-dependent situation of elasto-viscoplasticity was discussed. The elasto-plastic behaviour of plates in bending was considered in Chapter 9.

A.2.1 Program PLANET for the elasto-plastic analysis of plane and axisymmetric solids

CARD SET 1 TITLE CARD (12A6)—One card.

Cols. 1–72 Title of the problem—limited to 72 alphanumeric characters.

CARD SET 2 CONTROL CARD (11I5)—One card.

Cols. 1–5	NPOIN	Total number of nodal points.
6–10	NELEM	Total number of elements.
11–15	NVFIX	Total number of restrained boundary points—where one or more degrees of freedom are restrained.
16–20	NTYPE	Problem type parameter: 1—Plane stress, 2—Plane strain, 3—Axial symmetry.
21–25	NNODE	Number of nodes per element: 4—Linear quadrilateral element, 8—Quadratic Serendipity element, 9—Quadratic Lagrangian element.
26–30	NMATS	Total number of different materials.
31–35	NGAUS	Order of integration formula for numerical integration: 2—Two point Gauss quadrature rule, 3—Three point Gauss quadrature rule.

36–40	NALGO	Nonlinear solution parameter: 1 *Initial stiffness method.* The element stiffnesses are calculated at the beginning of the solution process and remain unchanged thereafter. 2 *Tangential stiffness method.* The element stiffnesses are recalculated for every iteration of each load increment. 3 *Combined algorithm* (*Version I*). The element stiffnesses are recalculated for the *first* iteration of each load increment only. 4 *Combined algorithm* (*Version II*). The element stiffnesses are recalculated for the *second* iteration of each load increment only.
41–45	NCRIT	Yield criterion parameter: 1—Tresca, 2—Von Mises, 3—Mohr–Coulomb, 4—Drucker–Prager.
46–50	NINCS	Number of increments in which the total loading is to be applied.
51–55	NSTRE	Number of stress components at a point: 3—Plane stress or plane strain, 4—Axial symmetry.

CARD SET 3 ELEMENT CARDS (11I5)—One card for each element. Total of NELEM cards (See Card Set 2).

Cols. 1–5	NUMEL	Element number.
6–10	MATNO(NUMEL)	Material property number.
11–15	LNODS(NUMEL,1)	1st Nodal connection number.
16–20	LNODS(NUMEL,2)	2nd Nodal connection number.
		. . . .
51–55	LNODS(NUMEL,9)	9th Nodal connection number.

Notes: 1) Columns 31–55 remain blank for linear 4-noded elements.

2) Columns 51–55 remain blank for 8-noded elements.

3) The nodal connection numbers must be listed in an anti-clockwise sequence, starting from any corner node.

CARD SET 4 NODE CARDS (I5,2F10.5)—One card for each node whose coordinates are to be input.

Cols. 1–5	IPOIN	Nodal point number.
6–15	COORD(IPOIN,1)	x (or r) coordinate of the node.
16–25	COORD(IPOIN,2)	y (or z) coordinate of the node.

Notes: 1) The total number of cards in this set will generally differ from NPOIN (see Card Set 2) since for quadratic elements whose sides are linear, it is only necessary to specify data for corner nodes, intermediate nodal coordinates being automatically interpolated if on a straight line.
2) For Lagrangian elements the coordinates of the 9th (central) node are never input.
3) The coordinates of the highest numbered node must be input regardless of whether it is a midside node or not.

CARD SET 5 RESTRAINED NODE CARDS (1X,I4,5X,I5,5X,2F10.5)—One card for each restrained node. Total of NVFIX cards (See Card Set 2).

Cols. 2–5	NOFIX(IVFIX)	Restrained node number.
11–15	IFPRE	Restraint code: 01 Nodal displacement restrained in the x (or r) direction, 10 Nodal displacement restrained in the y (or z) direction, 11 Nodal displacement restrained in both coordinate directions.
21–30	PRESC(IVFIX,1)	The prescribed value of the x (or r) component of nodal displacement.
31–40	PRESC(IVFIX,2)	The prescribed value of the y (or z) component of nodal displacement.

CARD SET 6 MATERIAL CARDS
6(a) CONTROL CARD (I5)—One card.

Cols. 1–5	NUMAT	Material identification number.

6(b) PROPERTIES CARDS (7F10.5)—One card for each different material.

Cols. 1–10	PROPS(NUMAT,1)	Elastic modulus, E.
11–20	PROPS(NUMAT,2)	Poisson's ratio, ν.
21–30	PROPS(NUMAT,3)	Material thickness, t (leave blank for plane strain and axisymmetric problems).
31–40	PROPS(NUMAT,4)	Mass density, ρ.
41–50	PROPS(NUMAT,5)	Uniaxial yield stress, σ_Y (or cohesion c for Mohr–Coulomb or Drucker–Prager materials).
51–60	PROPS(NUMAT,6)	Strain hardening parameter, H'.

61–70	PROPS(NUMAT,7)	Friction angle ϕ (measured in degrees) for Mohr–Coulomb and Drucker–Prager materials only).

Note: This card set to be repeated for each different material. Total of NMATS card sets (See Card Set 2).

CARD SET 7 LOAD CASE TITLE CARD (12A6)—One card.

Cols. 1–72	TITLE	Title of the load case—limited to 72 alphanumeric characters.

CARD SET 8 LOAD CONTROL CARD (3I5)—One card.

Cols. 1–5	IPLOD	Applied point load control parameter: 0 No applied nodal loads to be input, 1 Applied nodal loads to be input.
6–10	IGRAV	Gravity loading control parameter: 0 No gravity loads to be considered, 1 Gravity loading to be considered.
11–15	IEDGE	Distributed edge load control parameter: 0 No distributed edge loads to be input, 1 Distributed edge loads to be input.

CARD SET 9 APPLIED LOAD CARDS (I5,2F10.3)—One card for each loaded nodal point.

Cols. 1–5	LODPT	Node number.
6–15	POINT(1)	Load component in x (or r) direction.
16–25	POINT(2)	Load component in y (or z) direction.

Notes: 1) The last card should be that for the highest numbered node whether it is loaded or not.
2) For axisymmetric problems, the loads input should be the *total* loading on the circumferential ring passing through the nodal point concerned.
3) If IPLOD = 0 in Card Set 8, omit this set.

CARD SET 10 GRAVITY LOADING CARD (2F10.3)—One card.

Cols. 1–10	THETA	Angle of gravity axis measured from the positive y axis (see Fig. 6.7).
11–20	GRAVY	Gravity constant—specified as a multiple of the gravitational acceleration, g.

Note: If IGRAV = 0 in Card Set 8, omit this set.

CARD SET 11 DISTRIBUTED EDGE LOAD CARDS
11(a) CONTROL CARD (I5)—One card.

Cols. 1–5	NEDGE	Number of element edges on which distributed loads are to be applied.

11(b) ELEMENT FACE TOPOLOGY CARD (4I5)

Cols. 1–5	NEASS	The element number with which the element edge is associated.
6–10	NOPRS(1)	List of nodal points, in an anticlockwise sequence, of the nodes forming the element face on which the distributed load acts.
11–15	NOPRS(2)	
16–20	NOPRS(3)	

Note: For linear 4-noded elements, Cols. 16–20 remain blank.

11(c) DISTRIBUTED LOAD CARDS (6F10.3)

Cols. 1–10	PRESS(1,1)	Value of normal component of distributed load at node NOPRS(1).
11–20	PRESS(1,2)	Value of tangential component of distributed load at node NOPRS(1).
21–30	PRESS(2,1)	Value of normal component of distributed load at node NOPRS(2).
31–40	PRESS(2,2)	Value of tangential component of distributed load at node NOPRS(2).
41–50	PRESS(3,1)	Value of normal component of distributed load at node NOPRS(3).
51–60	PRESS(3,2)	Value of tangential component of distributed load at node NOPRS(3).

Notes: 1) For linear 4-noded elements, Cols. 41–60 remain blank.
2) Subsets 11(b) and 11(c) must be repeated in turn for every element edge on which a distributed load acts. The element edges can be considered in any order.
3) If IEDGE = 0 in Card Set 8, omit this card set.

CARD SET 12 LOAD INCREMENT CONTROL CARDS (2F10.5,3I5)—

One card for each load increment. Total of NINCS cards (see Card Set 2).

Cols. 1–10	FACTO	Applied load factor for this increment—specified as a factor of the loading input in Card Sets 8 to 11.
11–20	TOLER	Convergence tolerance factor.—The term TOLER in (3.27).
21–25	MITER	Maximum number of iterations allowed for the load increment.
26–30	NOUTP(1)	Parameter controlling output of results after 1st iteration: 0—No output, 1—Output displacements, 2—Output displacements and reactions,

		3—Output displacements, reactions and stresses.
31–35	NOUTP(2)	Parameter controlling output of the converged results: 0—No output, 1—Output displacements, 2—Output displacements and reactions, 3—Output displacements, reactions and stresses.

Note: The applied loading factors are accumulative. If FACTO is specified as 0.6, 0.3, 0.2 for the first three load increments, then the total loading acting during the third increment is 1.1 times that specified in Card Sets 8 to 11.

A.2.2 Program VISCOUNT for the elasto-viscoplastic analysis of plane and axisymmetric solids

The input data for this application is identical to that described in Section A.2.1, for elasto-plastic problems, with the following exceptions.

CARD SET 2 CONTROL CARD (11I5)

Cols. 36–40	NALGO	Equation solution parameter: 1 Explicit time stepping scheme (i.e. TIMEX = 0—See Card Set 12), 2 Implicit or Semi-implicit schemes (TIMEX $\neq$ 0).

CARD SET 6(b) PROPERTIES CARDS (8F10.5)—Two cards for each different material.

1st Card

Cols. 1–70		Identical to Card Set 6(b), Section A.2.1.
71–80	PROPS(NUMAT,8)	Fluidity parameter, γ.

2nd Card

Cols. 1–10	PROPS(NUMAT,9)	The constant M in (8.8) or constant N in (8.9).
11–20	PROPS(NUMAT,10)	Parameter controlling choice of the flow function: 0 Expression (8.8) to be used, 1 Expression (8.9) to be used.

CARD SET 12 TIMESTEPPING PARAMETER CARD (4F10.3)—One card.

Cols. 1–10	TIMEX	Timestepping algorithm parameter, Θ in (8.10).

11–20	TAUFT	The factor τ employed to limit the time step length according to (8.29).
21–30	DTINT	The initial time step length (required to initiate the time stepping process).
31–40	FTIME	The factor k in (8.32).

CARD SET 13 LOAD INCREMENT CONTROL CARDS

This card set is identical to Card Set 12, Section A.2.1 where the term 'iteration' is now replaced by 'timestep'.

A.2.3 Programs MINDLIN and MINDLAY for the nonlayered and layered elasto-plastic analysis of Mindlin plates

The input data for this application is identical to that described in Section A.2.1, for elasto-plastic plane and axisymmetric solids, with the following exceptions.

CARD SET 2 (11I5)—One card

Cols.16–20	NTYPE	Problem type parameter: 5—for Heterosis element, 0—for 4- or 8-node elements.
21–25	NNODE	Number of nodes per element: 4—Linear 4-node quadrilateral element. 8—Quadratic 8-node Serendipity element. 9—Quadratic 9-node Lagrangian element or Heterosis element.
31–35	NGAUS	2 for 4-node element, 3 for 8-, 9-node and Heterosis element. (N.B. This is the integration rule to evaluate the flexural contribution to the element stiffness matrix. Since selective integration is adopted a (NGAUS-1) integration is automatically used to evaluate the transverse shear contribution to the element stiffness matrix.)
41–45	NCRIT	Yield criterion parameter: 1—Tresca, 2—Von-Mises. (Mohr–Coulomb and Drucker–Prager yield criteria are not included.)
51–55	NLAPS	Total number of layers. (for program MINDLAY only—in program MINDLIN leave blank.)

CARD SET 5 RESTRAINED NODE CARDS (1X, I4, 5X, I5, 5X, 3F10.5)
One card for each restrained node. Total of NVFIX cards.

Cols.11–15	IFPRE	Restraint code: 100 Lateral displacement w restrained. 010 Rotation θ_x restrained. 001 Rotation θ_y restrained. 110 Lateral displacement w and rotation θ_x restrained, etc.
21–30	PRESC(IVFIX,1)	The prescribed value of the lateral nodal displacement w.
31–40	PRESC(IVFIX,2)	The prescribed value of the nodal rotation θ_x.
41–50	PRESC(IVFIX,3)	The prescribed value of the nodal rotation θ_y.

CARD SET 6 MATERIAL CARDS
6(b) PROPERTIES CARDS (7F10.5)—One card for each different material.

Cols.31–40	PROPS(NUMAT,4)	Uniform distributed loading value.
41–50	PROPS(NUMAT,5)	Blank.
51–60	PROPS(NUMAT,6)	Uniaxial yield stress, σ_0.
61–70	PROPS(NUMAT,7)	Strain hardening parameter H'.

CARD SET 6X CONVERGENCE CHECK CARDS
6X(a) DISPLACEMENT CHECK CARD (5I1)—One card.

Cols. 1	IFDIS	1	The displacement check is to be employed.
2	NCDIS(1)	1	Check based on norm involving w.
3	NCDIS(2)	1	Check based on norm involving θ_x.
4	NCDIS(3)	1	Check based on norm involving θ_y.
5	NCDIS(4)	1	Check based on w, θ_x and θ_y.

6X(b) RESIDUAL FORCE CHECK CARD (5I1)—One card.

Cols. 1	IFRES	1	The residual force check is to be employed.
2	NCRES(1)	1	Check based on norm involving residual forces associated with w.
3	NCRES(2)	1	Check based on norm involving residual forces associated with θ_x.
4	NCRES(3)	1	Check based on norm involving residual forces associated with θ_y.
5	NCRES(4)	1	Check based on norm involving residual forces associated with w, θ_x and θ_y.

Note: A zero value for any item implies that the check is not being used.

CARD SET 8 LOAD CONTROL CARD (I5)—One card.

Cols. 1–5	IPLOD	Applied point load control parameter: 0 No applied nodal loads to be input. 1 Applied nodal loads to be input.
6–15		Blank.

CARD SET 9 APPLIED LOAD CARDS (I5, 3F10.3)—One card for each loaded nodal point.

Cols. 1–5	LODPT	Node number.
6–15	POINT(1)	Lateral nodal load.
16–25	POINT(2)	Nodal couple in xz plane.
26–35	POINT(3)	Nodal couple in yz plane.

Omit CARD SETS 10, 11(a), 11(b) and 11(c).

Appendix III

Instructions for preparing input data for dynamic transient problems

The program DYNPAK has been described in Section 10.6 and MIXDYN in Section 11.5. These programs perform large displacement or viscoplastic or elasto-plastic, transient dynamic analysis of plane stress/strain or axi-symmetric problems respectively. The format of the input data is identical for both programs. In this appendix user instructions for preparing input data are provided.

CARD SET 1 DYNAMIC DIMENSIONING (4I5)—One card.

Cols.	1–5	NPOIN	Total number of nodal points.
	6–10	NELEM	Total number of elements.
	11–15	NDOFN	Number of degrees of freedom per node (= 2).
	16–20	NMATS	Number of different material sets.

CARD SET 2 TITLE CARD (10A4)—One card.

Cols.	1–40	Title of the problem—limited to 40 alphanumeric characters.

CARD SET 3 CONTROL CARD (13I5)—One card.

Cols.	1–5	NVFIX	Total number of nodal points with fixed degrees of freedom.
	6–10	NTYPE	Type of problem: = 1, Plane stress, = 2, Plane strain, = 3, Axisymmetric problem.
	11–15	NNODE	Number of nodes per element.
	16–20	NPROP	Number of material properties (= 11).
	21–25	NGAUS	Integration rule for stiffness matrix.
	26–30	NDIME	Number of coordinate dimensions (=2).
	31–35	NSTRE	Number of stress components (= 3 for plane stress/strain, = 4 for axisymmetric).

36–40	NCRIT	Yield criterion: = 1 — Tresca, = 2 — Von Mises, = 3 — Mohr–Coulomb, = 4 — Drucker–Prager.
41–45	NPREV	Indicator for the previous state to be read (= 1 for previous state, otherwise, = 0).
46–50	NCONM	Number of concentrated masses (⩾1 if concentrated mass present, otherwise, = 0).
51–55	NLAPS	Indicator for large displacement analysis: = 0—Elastic analysis, = 1—Elasto-plastic small displacement analysis, = 2—Elastic large displacement analysis,
56–60	NGAUM	Integration rule for mass matrix.
61–65	NRADS	= 0, Read (r, z) coordinates for nodes, = 1, Read (R, Θ) coordinates for nodes for axisymmetric analysis.

CARD SET 4 ELEMENT CARDS (11I5)—One card for each element, total of NELEM cards. The node numbers are read in anticlockwise sequence. The number of nodes depends upon the type of element. For four and eight noded elements read only four and eight nodes respectively.

Cols. 1–5	IELEM	Element number.
6–10	MATNO	Material identification number.
11–15	LNODS(IELEM,1)	
16–20	LNODS(IELEM,2)	
21–25	LNODS(IELEM,3)	
26–30	LNODS(IELEM,4)	
31–35	LNODS(IELEM,5)	Nodal connection numbers.
36–40	LNODS(IELEM,6)	
41–45	LNODS(IELEM,7)	
46–50	LNODS(IELEM,8)	
51–55	LNODS(IELEM,9)	

CARD SET 5 NODAL COORDINATE CARDS (I5,2F10.5)—One card for each node. Last nodal point (IPOIN = NPOIN) must be read at the end. Only corner and central nodes need to be specified. Midside nodes are interpolated if not specified. For axisymmetric cases, (R, Θ) values are read for NRADS = 1, and (r, z) coordinates are calculated in the program.

Cols. 1–5	IPOIN	Current nodal point.
6–15	COORD(IPOIN,1)	x-coordinate.*
16–25	COORD(IPOIN,2)	y-coordinate.

CARD SET 6 RESTRAINED NODE CARDS (1X,I4,3X,2I1)—One card for each restrained node. Total of NVFIX cards.

Cols. 2–5	IPOIN	Restrained node number.
9	IFPRE(IVFIX,1)	Fixity in x-direction (= 0, Free; = 1, Fixed).
10	IFPRE(IVFIX,2)	Fixity in y-direction (= 0, Free; = 1, Fixed).

CARD SET 7 MATERIAL CARDS—Three cards for each different material, a total of NMATS*3 cards.

1st Card MATERIAL IDENTIFICATION CARD (I5)

Cols. 1–5	NUMAT	Material identification number.

2nd Card MATERIAL PROPERTIES CARD—(a) (8E10.4)

Cols. 1–10 PROPS(NUMAT,1) Young's Modulus, E.

11–20 PROPS(NUMAT,2) Poisson's ratio, ν.

21–30 PROPS(NUMAT,3) Thickness for plane stress problem, t.

31–40 PROPS(NUMAT,4) Mass density per unit volume, ρ.

41–50 PROPS(NUMAT,5) Temperature coefficient, α_t.

51–60 PROPS(NUMAT,6) Reference yield value 'F_0':

Von Mises, $F_0 = \sigma_Y$,

Tresca, $F_0 = \sigma_Y$,

Mohr–Coulomb, $F_0 = c \cos \phi$,

Drucker–Prager, $F_0 = 6c \cos \phi / (\sqrt{3}(3-\sin \phi))$.

61–70 PROPS(NUMAT,7) Hardening parameter, H':

$$H' = \frac{E_T}{1-E_T/E},$$

where E_T is the hardening tangent modulus,

E is the tangent modulus,

σ_Y is the yield stress,

c is the cohesion,

ϕ is the friction angle.

71–80 PROPS(NUMAT,8) Friction angle 'ϕ'.

* For axisymmetric problems x and y are replaced by r and z respectively (or R and Θ if NRADS = 1).

3rd Card MATERIAL PROPERTIES CARD—(b) (3E10.4)

Cols. 1–10	PROPS(NUMAT,9)	Fluidity parameter, γ.
11–20	PROPS(NUMAT,10)	Exponent, δ.
21–30	PROPS(NUMAT,11)	NFLOW code (NFLOW = 1—Power law, NFLOW $\neq$ 1—Exponential law).

CARD SET 8 TIME INTEGRATION CONTROL CARD (11I5)—One card.

Cols. 1–5	NSTEP	Total number of time steps.
6–10	NOUTD	Writes displacement and stress history of required points on tapes 10 and 11 respectively at NOUTD timesteps.
11–15	NOUTP	Output for displacements and stresses at every NOUTP step (NOUTP $\leqslant$ 500).
16–20	NREQD	Number of nodes for selective output of displacements at NOUTD steps.
21–25	NREQS	Number of integration points for selective output of stresses at every NOUTP step.
26–30	NACCE	Number of acceleration ordinates (If IFUNC $\neq$ 0, NACCE is not used, then leave blank).
31–35	IFUNC	Time function code: IFUNC = 0 Acceleration time history, IFUNC = 1 Heaviside function, $f(t)$ = 1.0, IFUNC = 2 Harmonic excitation, $f(t) = a_0+b_0\sin\omega t$.
36–40	IFIXD	Indicator for excitation: IFIXD = 0, Horizontal acceleration read from tape 7, Vertical acceleration read from tape 12. IFIXD = 1, Vertical acceleration read from tape 12, IFIXD = 2, Horizontal acceleration read from tape 7. (If IFUNC $\neq$ 0 IFIXD is not used, then leave blank.)
41–45	MITER	Maximum number of iterations. This variable is not used in DYNPAK, so leave blank.

46–50	KSTEP	Number of steps after which the stiffness matrix is reformed. Not used in DYNPAK, leave blank.
51–55	IPRED	= 1 Standard algorithm, = 2 Modified algorithm.

CARD SET 9 TIME INTEGRATION PARAMETERS CARD (8F10.3)—Two cards.

1st Card

Cols. 1–10	DTIME	Time step length.
11–20	DTEND	Time at the end of the excitation force.
21–30	DTREC	Time step of acceleration records.
31–40	AALFA	α = Damping parameter, $C = \alpha M$, $\alpha = 2\xi_i\omega_i$.
41–50	BEETA	β = Damping parameter, $C = \beta K$. $(\alpha+\beta\omega_i^2 = 2\omega_i\xi_i$, not used in DYNPAK)
51–60	DELTA	Newmark's integration parameter ($\delta = 0.25\ (\gamma+0.5)^2$, not used in DYNPAK).
61–70	GAAMA	Newmark's integration parameter ($\gamma \geqslant 0.5$ for stable solution, not used in DYNPAK).
71–80	AZERO	Constants for harmonic excitation $f(t) = a_0 + b_0 \sin\omega t$.

2nd Card

1–10	BZERO	Constants for harmonic excitation $f(t) = a_0 + b_0 \sin\omega t$.
11–20	OMEGA	Constants for harmonic excitation $f(t) = a_0 + b_0 \sin\omega t$.
21–30	TOLER	Specified tolerance (Not used in DYNPAK).

CARD SET 10 CARD FOR NODAL POINTS FOR WHICH DISPLACEMENT HISTORY IS REQUIRED (16I5)—Total of NREQD nodes.

Cols. 1–5	NPRQD(1)	First nodal point at which displacement history is required.
6–10	NPRQD(2)	Second nodal point at which displacement history is required.
11–15	.	
.	.	
.	.	
.	.	

CARD SET 11 CARD FOR INTEGRATION POINTS FOR WHICH STRESS HISTORY IS REQUIRED (16I5)—Total of NREQS integration points.

Cols. 1–5	NGRQS(1)	First integration point at which stress history is required.
6–10	NGRQS(2)	Second integration point at which stress history is required.
11–15	.	
.	.	
.	.	
.	.	

CARD SET 12 IMPLICIT–EXPLICIT ELEMENT INDICATOR CARDS (16I5). Number of cards depends on number of elements. For each 16 elements one card is needed. In DYNPAK, INTGR(IELEM) is 2 for every element.

INTGR(IELEM) = 1, Implicit element.
INTGR(IELEM) = 2, Explicit element.

CARD SET 13 INITIAL DISPLACEMENT CARDS (I5,2F10.5)—One card for each node. If all displacements are zero, read data for last node.

Cols. 1–5	NGASH	Nodal point.
6–15	XGASH	Initial x-displacement.
16–25	YGASH	Initial y-displacement.

CARD SET 14 INITIAL VELOCITY CARDS (I5,2F10.5)—One card for each node. If all velocities are zero, read data for last node.

Cols. 1–5	NGASH	Nodal point.
6–15	XGASH	Initial x-velocity.
16–25	YGASH	Initial y-velocity.

CARD SET 15 PREVIOUS LOAD STATE CARDS (I5,2F10.3)—One card for one node, a total of NNODE cards. Data for the last nodal point should always be read even when it is not loaded. If NPREV = 0 then omit this set of data.

Cols. 1–5	NGASH	Nodal point.
6–15	XGASH	Equivalent nodal load in x direction.
16–25	YGASH	Equivalent nodal load in y direction.

CARD SET 16 PREVIOUS STRESS STATE CARD (I5,4F10.3)—One card for one integration point. Total of (NELEM*NGAUS*NGAUS) cards. If NPREV = 0 omit this set of data.

Cols. 1–5	KGAUS	Integration point.
6–15	STRESS(1)	Initial stress, σ_x or σ_r.
16–25	STRESS(2)	Initial stress, σ_y or σ_z.
26–35	STRESS(3)	Initial stress, γ_{xy} or γ_{rz}.
36–45	STRESS(4)	Initial stress, σ_z or σ_θ.

CARD SET 17 LOAD TITLE CARD (10A4)—One card.

Cols. 1–40 Title of load applied—limited to 40 alphanumeric characters.

CARD SET 18 LOAD INDICATOR CARD (4I5)—One card.

Cols. 1–5	IPLOD	Point load indicator.
6–10	IGRAV	Gravity load indicator.
11–15	IEDGE	Edge load indicator.
16–20	ITEMP	Temperature load indicator.

CARD SET 19 POINT LOAD CARD (I5,2F10.3)—One card for each node. Data for the last node must be specified at the end. If IPLOD = 0 then omit this set of data.

Cols. 1–5	LODPT	Node number.
6–15	POINT(1)	Load in x-direction.
16–25	POINT(2)	Load in y-direction.

CARD SET 20 GRAVITY LOAD CARD (2F10.3)—One card only. If IGRAV = 0 then omit this set of data.

Cols. 1–10	THETA	Angle of gravity axis to the positive y axis.
11–20	GRAVY	Gravity constant.

CARD SET 21 NUMBER OF PRESSURE EDGE CARD (I5)—One card. If IEDGE = 0, then omit card sets 21 and 22.

Cols. 1–5	NEDGE	Number of loaded edges.

CARD SET 22 PRESSURE CARDS—Two cards for each pressure loaded edge.

1st Card PRESSURE NODES CARD (4I5)—One card for each edge. Total of NEDGE cards.

Cols. 1–5	NEASS	Element number with edge load.
Cols. 6–10	NOPRS(1)	Edge nodes read in anticlockwise sequence.
11–15	NOPRS(2)	
16–20	NOPRS(3)	

2nd Card PRESSURE CARD (6F10.3)—One card for each edge. Total of NEDGE cards. A pressure normal to a face is assumed to be positive if it acts in a direction into the element. A tangential load is assumed to be positive if it acts in an anticlockwise direction with respect to the loauedWW positive if it acts in an anticlockwise direction with respect to the loaded element.

Cols.	1–10	PRESS(1,1)	Normal component of edge load for each node.
	11–20	PRESS(2,1)	
	21–30	PRESS(3,1)	
	31–40	PRESS(1,2)	Tangential component of edge load for each node.
	41–50	PRESS(2,2)	
	51–60	PRESS(3,2)	

CARD SET 24 TEMPERATURE CARDS (I5, F10.3)—One card for each node. The last card must be for the highest numbered node. If ITEMP = 0, omit this set of data.

Cols.	1–5	NODPT	Node number.
	6–15	TEMPE	Nodal temperature.

CARD SET 25 CONCENTRATED MASSES (I5,2F10.3)—One card for each node. Total of NCONM cards. If NCONM = 0, omit this set of data.

Cols.	1–5	IPOIN	Current nodal point with concentrated mass.
	6–15	XCMAS	Concentrated mass associated with the x-direction.
	16–25	YCMAS	Concentrated mass associated with the y-direction.

Appendix IV

Sample input data and line printer output for one – and two-dimensional applications

In this appendix input data and line printer output are provided for a selection of the numerical examples presented in the text. This information will be of assistance to readers who wish to implement the programs contained in the book on their own computer. For economy of space, presentation is limited to one example from each area of application. Also in some cases the line printer output is edited for the same reason.

A.4.1 Solution of one-dimensional quasiharmonic problem by direct iteration. Example of Section 3.9.3, Fig. 3.3

Input data

```
     1-D QUASIHARMONIC EXAMPLE , SECTION 3.9.3 , FIG. 3.3
11   10    2    1    1    2    1    1    1
      1        10.0
 1    1    2    1
 2    2    3    1
 3    3    4    1
 4    4    5    1
 5    5    6    1
 6    6    7    1
 7    7    8    1
 8    8    9    1
 9    9   10    1
10   10   11    1
      1   0.0
      2   1.0
      3   2.0
      4   3.0
      5   4.0
      6   5.0
      7   6.0
      8   7.0
      9   8.0
     10   9.0
     11  10.0
      1    1   0.0
     11    1   1.0
     10         0.0          0.0
20    1         1.0          0.5
```

Line printer output

```
    1-D QUASIHARMONIC EXAMPLE , SECTION 3.9.3 , FIG. 3.3
NPOIN =   11   NELEM =   10   NBOUN =    2   NMATS =    1
NPROP =    1   NNODE =    2   NINCS =    1   NALGO =    1
NDOFN =    1
    MATERIAL PROPERTIES
       1       10.00000
  EL   NODES   MAT.
   1    1   2    1
   2    2   3    1
   3    3   4    1
   4    4   5    1
   5    5   6    1
   6    6   7    1
   7    7   8    1
   8    8   9    1
   9    9  10    1
  10   10  11    1
    NODE      COORD.
       1        0.00000
       2        1.00000
       3        2.00000
       4        3.00000
       5        4.00000
       6        5.00000
       7        6.00000
       8        7.00000
       9        8.00000
      10        9.00000
      11       10.00000
 RES.NODE  CODE   PRES.VALUES
       1     1        0.00000
      11     1        1.00000
  ELEMENT            NODAL LOADS
       1        0.00000        0.00000
       2        0.00000        0.00000
       3        0.00000        0.00000
       4        0.00000        0.00000
       5        0.00000        0.00000
       6        0.00000        0.00000
       7        0.00000        0.00000
       8        0.00000        0.00000
       9        0.00000        0.00000
      10        0.00000        0.00000
    IINCS =    1   NITER =   20   NOUTP =    1   FACTO =  0.100000E 01   TOLER =  0.500000E 00
    CONVERGENCE CODE =    1   NORM OF RESIDUAL SUM RATIO =  0.000000E 00
    NODE    DISPL.            REACTIONS
       1  0.000000E 00      -0.100000E 01
       2  0.100000E 00       0.000000E 00
       3  0.200000E 00       0.000000E 00
       4  0.300000E 00       0.000000E 00
       5  0.400000E 00       0.000000E 00
       6  0.500000E 00       0.000000E 00
       7  0.600000E 00       0.000000E 00
       8  0.700000E 00       0.000000E 00
       9  0.800000E 00       0.000000E 00
      10  0.900000E 00       0.000000E 00
      11  0.100000E 01       0.100000E 01
  ELEMENT      STRESSES       PL.STRAIN
       1  0.000000E 00  0.000000E 00
       2  0.000000E 00  0.000000E 00
       3  0.000000E 00  0.000000E 00
       4  0.000000E 00  0.000000E 00
       5  0.000000E 00  0.000000E 00
```

```
      6  0.000000E 00  0.000000E 00
      7  0.000000E 00  0.000000E 00
      8  0.000000E 00  0.000000E 00
      9  0.000000E 00  0.000000E 00
     10  0.000000E 00  0.000000E 00
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.706275E 02
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.393376E 02
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.983804E 01
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.801219E 01
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.472308E 01
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.127390E 01
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.974302E 00
   CONVERGENCE CODE =  1   NORM OF RESIDUAL SUM RATIO =  0.574815E 00
   CONVERGENCE CODE =  0   NORM OF RESIDUAL SUM RATIO =  0.153335E 00
   NODE      DISPL.             REACTIONS
      1  0.000000E 00       -0.600000E 01
      2  0.260555E 00        0.000000E 00
      3  0.399999E 00        0.000000E 00
      4  0.508276E 00        0.000000E 00
      5  0.599999E 00        0.000000E 00
      6  0.681025E 00        0.000000E 00
      7  0.754400E 00        0.000000E 00
      8  0.821954E 00        0.000000E 00
      9  0.884886E 00        0.000000E 00
     10  0.944031E 00        0.000000E 00
     11  0.100000E 01        0.600000E 01
ELEMENT      STRESSES      PL.STRAIN
      1  0.000000E 00  0.000000E 00
      2  0.000000E 00  0.000000E 00
      3  0.000000E 00  0.000000E 00
      4  0.000000E 00  0.000000E 00
      5  0.000000E 00  0.000000E 00
      6  0.000000E 00  0.000000E 00
      7  0.000000E 00  0.000000E 00
      8  0.000000E 00  0.000000E 00
      9  0.000000E 00  0.000000E 00
     10  0.000000E 00  0.000000E 00
```

A.4.2 Solution of one-dimensional elasto-plastic problem. Example of Section 3.12.3, Fig. 3.9

Input data

```
  1-D ELASTO-PLASTIC EXAMPLE , SECTION 3.12.3 ,FIG. 3.9
11   10    2    2    4    2   16    3    1
      1   10000.0            1.0            5.0            1000.0
      2   10000.0            2.0            7.5            2000.0
 1    1    2    1
 2    2    3    1
 3    3    4    1
 4    4    5    1
 5    5    6    1
 6    6    7    2
 7    7    8    2
 8    8    9    2
 9    9   10    2
10   10   11    2
      1        0.0
      2        1.0
      3        2.0
      4        3.0
```

```
      5         4.0
      6         5.0
      7         4.0
      8         3.0
      9         2.0
     10         1.0
     11         0.0
      1    1    0.0
     11    1    0.0
      5         0.0            10.0
     10         0.0             0.0
30    2         1.25            0.5
30    2         0.25            0.5
30    2         0.25            0.5
30    2         0.25            0.5
30    2         0.25            0.5
30    2         0.25            0.5
30    2        -3.5             0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
30    2        -0.25            0.5
```

Line printer output

```
     1-D ELASTO-PLASTIC EXAMPLE , SECTION 3.12.3 ,FIG. 3.9
NPOIN =    11   NELEM =    10   NBOUN =     2   NMATS =     2
NPROP =     4   NNODE =     2   NINCS =    16   NALGO =     3
NDOFN =     1
     MATERIAL PROPERTIES
         1    10000.00000        1.00000        5.00000      1000.00000
         2    10000.00000        2.00000        7.50000      2000.00000
   EL    NODES   MAT.
    1    1    2    1
    2    2    3    1
    3    3    4    1
    4    4    5    1
    5    5    6    1
    6    6    7    2
    7    7    8    2
    8    8    9    2
    9    9   10    2
   10   10   11    2
     NODE      COORD.
         1          0.00000
         2          1.00000
         3          2.00000
         4          3.00000
         5          4.00000
         6          5.00000
         7          4.00000
         8          3.00000
         9          2.00000
        10          1.00000
        11          0.00000
 RES.NODE   CODE    PRES.VALUES
         1     1        0.00000
        11     1        0.00000
```

```
ELEMENT          NODAL LOADS
     1        0.00000         0.00000
     2        0.00000         0.00000
     3        0.00000         0.00000
     4        0.00000         0.00000
     5        0.00000        10.00000
     6        0.00000         0.00000
     7        0.00000         0.00000
     8        0.00000         0.00000
     9        0.00000         0.00000
    10        0.00000         0.00000
   IINCS =    1   NITER =   30   NOUTP =    2   FACTO =  0.125000E 01   TOLER =  0.500000E 00
   ITERATION NUMBER =    1
   CONVERGENCE CODE =   0   NORM OF RESIDUAL SUM RATIO =  0.629197E-08
   NODE    DISPL.            REACTIONS
     1  0.000000E 00       -0.416667E 01
     2  0.416667E-03        0.000000E 00
     3  0.833333E-03        0.000000E 00
     4  0.125000E-02        0.000000E 00
     5  0.166667E-02        0.000000E 00
     6  0.208333E-02        0.000000E 00
     7  0.166667E-02        0.000000E 00
     8  0.125000E-02        0.000000E 00
     9  0.833333E-03        0.000000E 00
    10  0.416667E-03        0.000000E 00
    11  0.000000E 00       -0.833333E 01
ELEMENT     STRESSES     PL.STRAIN
     1  0.416667E 01  0.000000E 00
     2  0.416667E 01  0.000000E 00
     3  0.416667E 01  0.000000E 00
     4  0.416667E 01  0.000000E 00
     5  0.416667E 01  0.000000E 00
     6  0.416667E 01  0.000000E 00
     7  0.416667E 01  0.000000E 00
     8  0.416667E 01  0.000000E 00
     9  0.416667E 01  0.000000E 00
    10  0.416667E 01  0.000000E 00

        .            .            .            .
        .            .            .            .
        .            .            .            .
        .            .            .            .

   IINCS =    3   NITER =   30   NOUTP =    2   FACTO =  0.250000E 00   TOLER =  0.500000E 00
   ITERATION NUMBER =    1
   CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.490863E 01
   NODE    DISPL.            REACTIONS
     1  0.000000E 00       -0.583333E 01
     2  0.583333E-03        0.000000E 00
     3  0.116667E-02        0.000000E 00
     4  0.175000E-02        0.000000E 00
     5  0.233333E-02        0.000000E 00
     6  0.291667E-02        0.000000E 00
     7  0.233333E-02        0.000000E 00
     8  0.175000E-02        0.000000E 00
     9  0.116667E-02        0.000000E 00
    10  0.583333E-03        0.000000E 00
    11  0.000000E 00       -0.116667E 02
ELEMENT     STRESSES     PL.STRAIN
     1  0.507576E 01  0.757576E-04
     2  0.507576E 01  0.757576E-04
     3  0.507576E 01  0.757576E-04
     4  0.507576E 01  0.757576E-04
```

```
     5  0.507576E 01  0.757576E-04
     6  0.583333E 01  0.000000E 00
     7  0.583333E 01  0.000000E 00
     8  0.583333E 01  0.000000E 00
     9  0.583333E 01  0.000000E 00
    10  0.583333E 01  0.000000E 00
   ITERATION NUMBER =     2
   CONVERGENCE CODE =    1   NORM OF RESIDUAL SUM RATIO =  0.147757E 01
   NODE     DISPL.             REACTIONS
     1  0.000000E 00       -0.532828E 01
     2  0.608586E-03        0.000000E 00
     3  0.121717E-02        0.000000E 00
     4  0.182576E-02        0.000000E 00
     5  0.243434E-02        0.000000E 00
     6  0.304293E-02        0.000000E 00
     7  0.243434E-02        0.000000E 00
     8  0.182576E-02        0.000000E 00
     9  0.121717E-02        0.000000E 00
    10  0.608586E-03        0.000000E 00
    11  0.000000E 00       -0.121717E 02
ELEMENT      STRESSES      PL.STRAIN
     1  0.509871E 01  0.987144E-04
     2  0.509871E 01  0.987144E-04
     3  0.509871E 01  0.987144E-04
     4  0.509871E 01  0.987144E-04
     5  0.509871E 01  0.987144E-04
     6  0.608586E 01  0.000000E 00
     7  0.608586E 01  0.000000E 00
     8  0.608586E 01  0.000000E 00
     9  0.608586E 01  0.000000E 00
    10  0.608586E 01  0.000000E 00
   ITERATION NUMBER =     3
   CONVERGENCE CODE =    0   NORM OF RESIDUAL SUM RATIO =  0.446758E 00
   NODE     DISPL.             REACTIONS
     1  0.000000E 00       -0.517524E 01
     2  0.616238E-03        0.000000E 00
     3  0.123248E-02        0.000000E 00
     4  0.184871E-02        0.000000E 00
     5  0.246495E-02        0.000000E 00
     6  0.308119E-02        0.000000E 00
     7  0.246495E-02        0.000000E 00
     8  0.184871E-02        0.000000E 00
     9  0.123248E-02        0.000000E 00
    10  0.616238E-03        0.000000E 00
    11  0.000000E 00       -0.123248E 02
ELEMENT      STRESSES      PL.STRAIN
     1  0.510567E 01  0.105671E-03
     2  0.510567E 01  0.105671E-03
     3  0.510567E 01  0.105671E-03
     4  0.510567E 01  0.105671E-03
     5  0.510567E 01  0.105671E-03
     6  0.616238E 01  0.000000E 00
     7  0.616238E 01  0.000000E 00
     8  0.616238E 01  0.000000E 00
     9  0.616238E 01  0.000000E 00
    10  0.616238E 01  0.000000E 00

     .          .           .           .
     .          .           .           .
     .          .           .           .
     .          .           .           .
               etc.
```

A.4.3 Solution of one-dimensional elasto-viscoplastic problem. Example of Section 4.12, Fig. 4.6

Input data

```
   1-D ELASTO VISCO-PLASTIC EXAMPLE , SECTION 4.12 , FIG. 4.6
 2    1    1    1    5    2    1    3    1
 1    10000.0          1.0          10.0         5000.0           0.001
 1    1    2    1
      1       0.0
      2      10.0
      1    1  0.0
      1       0.0          15.0
    0.05          0.025          1.5
90    2       1.0          0.1
```

Line printer output

```
    1-D ELASTO VISCO-PLASTIC EXAMPLE , SECTION 4.12 , FIG. 4.6
NPOIN =     2   NELEM =     1   NBOUN =     1   NMATS =     1
NPROP =     5   NNODE =     2   NINCS =     1   NALGO =     3
NDOFN =     1
    MATERIAL PROPERTIES
   1   10000.00000        1.00000       10.00000      5000.00000        0.00100
   EL   NODES   MAT.
   1    1    2    1
    NODE     COORD.
       1         0.00000
       2        10.00000
 RES.NODE  CODE   PRES.VALUES
       1     1        0.00000
  ELEMENT          NODAL LOADS
       1         0.00000        15.00000
    TAUFT =  0.500000E-01   DTINT =  0.250000E-01   FTIME =  0.150000E 01
    IINCS =     1   NSTEP =   90   NOUTP =    2   FACTO =  0.100000E 01   TOLER =  0.100000E 00
    TOTAL TIME =     0.000000E 00
    CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.100000E 03
    NODE    DISPL.              REACTIONS
       1  0.000000E 00      -0.150000E 02
       2  0.150000E-01       0.000000E 00
  ELEMENT     STRESSES     PL.STRAIN
       1  0.150000E 02  0.000000E 00
    TOTAL TIME =     0.250000E-01
    CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.650000E 02
    NODE    DISPL.              REACTIONS
       1  0.000000E 00      -0.150000E 02
       2  0.162500E-01       0.000000E 00
  ELEMENT     STRESSES     PL.STRAIN
       1  0.150000E 02  0.125000E-03
    TOTAL TIME =     0.435714E-01
    CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.682500E 02
    NODE    DISPL.              REACTIONS
       1  0.000000E 00      -0.150000E 02
       2  0.170625E-01       0.000000E 00
  ELEMENT     STRESSES     PL.STRAIN
       1  0.150000E 02  0.206250E-03
    TOTAL TIME =     0.650675E-01
    CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.716625E 02
    NODE    DISPL.              REACTIONS
       1  0.000000E 00      -0.150000E 02
       2  0.179156E-01       0.000000E 00
```

```
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.291562E-03
   TOTAL TIME =      0.903564E-01
   CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.752456E 02
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.188114E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.381141E-03
   TOTAL TIME =      0.120753E 00
   CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.790079E 02
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.197520E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.475198E-03
   TOTAL TIME =      0.158390E 00
   CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.829583E 02
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.207396E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.573958E-03
   TOTAL TIME =      0.207070E 00
   CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.871062E 02
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.217766E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.677655E-03
   TOTAL TIME =      0.274627E 00
   CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.865247E 02
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.228654E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.786538E-03
   TOTAL TIME =      0.375962E 00
   CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.640271E 02
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.239469E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.894694E-03
   TOTAL TIME =      0.527964E 00
   CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.230485E 02
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.247473E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.974728E-03
   TOTAL TIME =      0.755969E 00
   CONVERGENCE CODE =   0   NORM OF RESIDUAL SUM RATIO =  0.000000E 00
   NODE     DISPL.               REACTIONS
      1  0.000000E 00        -0.150000E 02
      2  0.250354E-01         0.000000E 00
ELEMENT      STRESSES       PL.STRAIN
      1  0.150000E 02  0.100354E-02
```

A.4.4 Solution of elasto-plastic layered Timoshenko beam. Example of Section 5.5.6, Fig. 5.11

Input data

```
1-D  EP TIMOSHENKO LAYERED BEAM EXAMPLE , SECTION 5.5.6 , FIG. 5.11
 11   10    2    1   17    2   14    2    2    6
  1
        210.0          53.8444          0.25000           0.0
        200.0           200.0            20.0             10.0
         40.0            10.0            40.0             10.0
         40.0            10.0            40.0            200.0
         20.0
  1    1    2    1
  2    2    3    1
  3    3    4    1
  4    4    5    1
  5    5    6    1
  6    6    7    1
  7    7    8    1
  8    8    9    1
  9    9   10    1
 10   10   11    1
       1             0.0
       2           300.0
       3           600.0
       4           900.0
       5          1200.0
       6          1500.0
       7          1800.0
       8          2100.0
       9          2400.0
      10          2700.0
      11          3000.0
       1    1                 0.0     1              0.0
      11    1                 0.0     1              0.0
       1        68.85000        0.00000        68.85000         0.00000
       2        68.85000        0.00000        68.85000         0.00000
       3        68.85000        0.00000        68.85000         0.00000
       4        68.85000        0.00000        68.85000         0.00000
       5        68.85000        0.00000        68.85000         0.00000
       6        68.85000        0.00000        68.85000         0.00000
       7        68.85000        0.00000        68.85000         0.00000
       8        68.85000        0.00000        68.85000         0.00000
       9        68.85000        0.00000        68.85000         0.00000
      10        68.85000        0.00000        68.85000         0.00000
100    2            0.30           0.50
100    2            0.20           0.50
100    2            0.10           0.50
100    2            0.10           0.50
100    2            0.05           0.50
100    2            0.05           0.50
100    2            0.05           0.50
100    2            0.05           0.50
100    2            0.02           0.50
100    2            0.02           0.50
100    2            0.02           0.50
100    2            0.02           0.50
100    2            0.01           0.50
100    2            0.01           0.50
```

Line printer output

```
 1-D  EP TIMOSHENKO LAYERED BEAM EXAMPLE , SECTION 5.5.6 , FIG. 5.11


NPOIN =   11   NELEM =   10   NBOUN =    2   NMATS =    1

NPROP =   17   NNODE =    2   NINCS =   14   NALGO =    2

NDOFN =    2   NLAYR =    6
     MATERIAL PROPERTIES
   1
     210.00000        53.84440         0.25000         0.00000
     200.00000       200.00000        20.00000        10.00000
      40.00000        10.00000        40.00000        10.00000
      40.00000        10.00000        40.00000       200.00000
      20.00000
   EL    NODES    MAT.
   1     1    2     1
   2     2    3     1
   3     3    4     1
   4     4    5     1
   5     5    6     1
   6     6    7     1
   7     7    8     1
   8     8    9     1
   9     9   10     1
  10    10   11     1
     NODE      COORD.
        1          0.00000
        2        300.00000
        3        600.00000
        4        900.00000
        5       1200.00000
        6       1500.00000
        7       1800.00000
        8       2100.00000
        9       2400.00000
       10       2700.00000
       11       3000.00000

 RES.NODE  CODE    PRES.VALUES  CODE    PRES.VALUES
        1    1         0.00000    1         0.00000
       11    1         0.00000    1         0.00000
  ELEMENT           NODAL LOADS
        1       68.85000         0.00000        68.85000         0.00000
        2       68.85000         0.00000        68.85000         0.00000
        3       68.85000         0.00000        68.85000         0.00000
        4       68.85000         0.00000        68.85000         0.00000
        5       68.85000         0.00000        68.85000         0.00000
        6       68.85000         0.00000        68.85000         0.00000
        7       68.85000         0.00000        68.85000         0.00000
        8       68.85000         0.00000        68.85000         0.00000
        9       68.85000         0.00000        68.85000         0.00000
       10       68.85000         0.00000        68.85000         0.00000
     IINCS =     1   NITER =  100   NOUTP =     2   FACTO =  0.300000E 00   TOLER =  0.500000E 00
     ITERATION NUMBER =     1
     CONVERGENCE CODE =   0   NORM OF RESIDUAL SUM RATIO =  0.113611E-07
     NODE         DISPLACEMENTS                              REACTIONS
        1  0.000000E 00      -0.206550E 03  0.000000E 00      -0.102242E 06
        2  0.342210E 00       0.000000E 00  0.156214E-02       0.000000E 00
        3  0.972874E 00       0.000000E 00  0.208286E-02       0.000000E 00
        4  0.161862E 01       0.000000E 00  0.182250E-02       0.000000E 00
        5  0.208417E 01       0.000000E 00  0.104143E-02       0.000000E 00
```

```
   6  0.225237E 01        0.000000E 00 -0.255548E-12       0.000000E 00
   7  0.208417E 01        0.000000E 00 -0.104143E-02       0.000000E 00
   8  0.161862E 01        0.000000E 00 -0.182250E-02       0.000000E 00
   9  0.972874E 00        0.000000E 00 -0.208286E-02       0.000000E 00
  10  0.342210E 00        0.000000E 00 -0.156214E-02       0.000000E 00
  11  0.000000E 00       -0.206550E 03  0.000000E 00       0.102242E 06
ELEMENT            STRESSES
   1 -0.743580E 05  0.185895E 03
   2 -0.247860E 05  0.144585E 03
   3  0.123930E 05  0.103275E 03
   4  0.371790E 05  0.619650E 02
   5  0.495720E 05  0.206550E 02
   6  0.495720E 05 -0.206550E 02
   7  0.371790E 05 -0.619650E 02
   8  0.123930E 05 -0.103275E 03
   9 -0.247860E 05 -0.144585E 03
  10 -0.743580E 05 -0.185895E 03

      .          .          .          .
      .          .          .          .
      .          .          .          .
      .          .          .          .

 IINCS =    6   NITER =  100   NOUTP =    2   FACTO =  0.500000E-01   TOLER =  0.500000E 00
 ITERATION NUMBER =     1
 CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.464588E 01
 NODE         DISPLACEMENTS                              REACTIONS
   1  0.000000E 00       -0.550800E 03  0.000000E 00      -0.272646E 06
   2  0.912561E 00        0.000000E 00  0.416571E-02       0.000000E 00
   3  0.259433E 01        0.000000E 00  0.555429E-02       0.000000E 00
   4  0.431631E 01        0.000000E 00  0.486000E-02       0.000000E 00
   5  0.555778E 01        0.000000E 00  0.277714E-02       0.000000E 00
   6  0.600632E 01        0.000000E 00 -0.645258E-13       0.000000E 00
   7  0.555778E 01        0.000000E 00 -0.277714E-02       0.000000E 00
   8  0.431631E 01        0.000000E 00 -0.486000E-02       0.000000E 00
   9  0.259433E 01        0.000000E 00 -0.555429E-02       0.000000E 00
  10  0.912561E 00        0.000000E 00 -0.416571E-02       0.000000E 00
  11  0.000000E 00       -0.550800E 03  0.000000E 00       0.272646E 06
ELEMENT            STRESSES
   1 -0.189331E 06  0.495720E 03
   2 -0.660960E 05  0.385560E 03
   3  0.330480E 05  0.275400E 03
   4  0.991440E 05  0.165240E 03
   5  0.132192E 06  0.550800E 02
   6  0.132192E 06 -0.550800E 02
   7  0.991440E 05 -0.165240E 03
   8  0.330480E 05 -0.275400E 03
   9 -0.660960E 05 -0.385560E 03
  10 -0.189331E 06 -0.495720E 03
 ITERATION NUMBER =     2
 CONVERGENCE CODE =   0   NORM OF RESIDUAL SUM RATIO =  0.210144E-08
 NODE         DISPLACEMENTS                              REACTIONS
   1  0.000000E 00       -0.550800E 03  0.000000E 00      -0.265108E 06
   2  0.100758E 01        0.000000E 00  0.479915E-02       0.000000E 00
   3  0.285562E 01        0.000000E 00  0.602936E-02       0.000000E 00
   4  0.469637E 01        0.000000E 00  0.517672E-02       0.000000E 00
   5  0.600911E 01        0.000000E 00  0.293550E-02       0.000000E 00
   6  0.648140E 01        0.000000E 00 -0.118097E-12       0.000000E 00
   7  0.600911E 01        0.000000E 00 -0.293550E-02       0.000000E 00
   8  0.469637E 01        0.000000E 00 -0.517672E-02       0.000000E 00
   9  0.285562E 01        0.000000E 00 -0.602936E-02       0.000000E 00
```

```
   10  0.100758E 01        0.000000E 00 -0.479915E-02        0.000000E 00
   11  0.000000E 00       -0.550800E 03  0.000000E 00        0.265108E 06
ELEMENT            STRESSES
    1 -0.190750E 06  0.495720E 03
    2 -0.585581E 05  0.385560E 03
    3  0.405859E 05  0.275400E 03
    4  0.106682E 06  0.165240E 03
    5  0.139730E 06  0.550800E 02
    6  0.139730E 06 -0.550800E 02
    7  0.106682E 06 -0.165240E 03
    8  0.405859E 05 -0.275400E 03
    9 -0.585581E 05 -0.385560E 03
   10 -0.190750E 06 -0.495720E 03

       .          .          .          .
       .          .          .          .
       .          .          .          .
       .          .          .          .

  IINCS =    11   NITER =  100   NOUTP =     2   FACTO =  0.200000E-01   TOLER =  0.500000E 00
  ITERATION NUMBER =     1
  CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.149229E 01
  NODE          DISPLACEMENTS                         REACTIONS
    1  0.000000E 00       -0.660960E 03  0.000000E 00       -0.287981E 06
    2  0.486620E 01        0.000000E 00  0.301397E-01        0.000000E 00
    3  0.143031E 02        0.000000E 00  0.309826E-01        0.000000E 00
    4  0.235411E 02        0.000000E 00  0.293260E-01        0.000000E 00
    5  0.319556E 02        0.000000E 00  0.260032E-01        0.000000E 00
    6  0.358944E 02        0.000000E 00  0.210285E-09        0.000000E 00
    7  0.319556E 02        0.000000E 00 -0.260032E-01        0.000000E 00
    8  0.235411E 02        0.000000E 00 -0.293260E-01        0.000000E 00
    9  0.143031E 02        0.000000E 00 -0.309826E-01        0.000000E 00
   10  0.486620E 01        0.000000E 00 -0.301397E-01        0.000000E 00
   11  0.000000E 00       -0.660960E 03  0.000000E 00        0.287981E 06
ELEMENT            STRESSES
    1 -0.196000E 06  0.594864E 03
    2 -0.401209E 05  0.462672E 03
    3  0.788519E 05  0.330480E 03
    4  0.158167E 06  0.198288E 03
    5  0.196000E 06  0.660960E 02
    6  0.196000E 06 -0.660960E 02
    7  0.158167E 06 -0.198288E 03
    8  0.788519E 05 -0.330480E 03
    9 -0.401209E 05 -0.462672E 03
   10 -0.196000E 06 -0.594864E 03
  ITERATION NUMBER =     2
  CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.562938E 10
  NODE          DISPLACEMENTS                         REACTIONS
    1  0.000000E 00       -0.656460E 03  0.000000E 00       -0.284525E 06
    2 -0.227149E 08        0.000000E 00 -0.151432E 06        0.000000E 00
    3 -0.681446E 08        0.000000E 00 -0.151432E 06        0.000000E 00
    4 -0.113574E 09        0.000000E 00 -0.151432E 06        0.000000E 00
    5 -0.159004E 09        0.000000E 00 -0.151432E 06        0.000000E 00
    6 -0.163102E 09        0.000000E 00  0.124115E 06        0.000000E 00
    7 -0.126424E 09        0.000000E 00  0.120404E 06        0.000000E 00
    8 -0.903028E 08        0.000000E 00  0.120404E 06        0.000000E 00
    9 -0.541817E 08        0.000000E 00  0.120404E 06        0.000000E 00
   10 -0.180606E 08        0.000000E 00  0.120404E 06        0.000000E 00
   11  0.000000E 00       -0.656351E 03  0.000000E 00        0.284576E 06
ELEMENT            STRESSES
    1  0.719122E 13  0.589934E 03
    2 -0.390314E 05  0.457742E 03
```

```
      3  0.784888E 05  0.327522E 03
      4  0.156624E 06  0.197302E 03
      5 -0.131161E 14  0.684992E 02
      6  0.196000E 06 -0.616594E 02
      7  0.156896E 06 -0.197302E 03
      8  0.787157E 05 -0.325057E 03
      9 -0.388044E 05 -0.458235E 03
     10  0.573122E 13 -0.590427E 03
   ITERATION NUMBER =     3
   CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.247769E 12
   NODE         DISPLACEMENTS                           REACTIONS
      1  0.000000E 00          0.386547E 11  0.000000E 00         0.131941E 14
      2 -0.256689E 18          0.000000E 00 -0.171126E 16         0.000000E 00
      3 -0.770066E 18          0.000000E 00 -0.171126E 16         0.000000E 00
      4 -0.128344E 19          0.000000E 00 -0.171126E 16         0.000000E 00
      5 -0.179682E 19          0.000000E 00 -0.171126E 16         0.000000E 00
      6 -0.707142E 18          0.000000E 00  0.897579E 16         0.000000E 00
      7  0.559323E 18          0.000000E 00 -0.532688E 15         0.000000E 00
      8  0.399516E 18          0.000000E 00 -0.532688E 15         0.000000E 00
      9  0.239710E 18          0.000000E 00 -0.532688E 15         0.000000E 00
     10  0.799033E 17          0.000000E 00 -0.532688E 15         0.000000E 00
     11  0.000000E 00          0.316249E 09  0.000000E 00        -0.594731E 13
ELEMENT             STRESSES
      1  0.719122E 13 -0.381105E 11
      2 -0.195980E 06 -0.169380E 11
      3 -0.195887E 06 -0.846899E 10
      4 -0.195820E 06  0.197302E 03
      5 -0.131161E 14  0.684992E 02
      6  0.196000E 06 -0.616594E 02
      7  0.196011E 06  0.148207E 11
      8  0.195954E 06  0.211725E 10
      9  0.195971E 06  0.635174E 10
     10 -0.253560E 23  0.211725E 10
   ITERATION NUMBER =     4
   CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.576146E 14
   NODE         DISPLACEMENTS                           REACTIONS
      1  0.000000E 00          0.386547E 11  0.000000E 00         0.131941E 14
      2  0.808314E 27          0.000000E 00  0.538876E 25         0.000000E 00
      3  0.808244E 27          0.000000E 00 -0.538923E 25         0.000000E 00
      4 -0.940584E 28          0.000000E 00 -0.627047E 26         0.000000E 00
      5 -0.116832E 25          0.000000E 00  0.125402E 27         0.000000E 00
      6  0.679753E 25          0.000000E 00 -0.125349E 27         0.000000E 00
      7 -0.395493E 26          0.000000E 00  0.125040E 27         0.000000E 00
      8  0.230105E 27          0.000000E 00 -0.123243E 27         0.000000E 00
      9 -0.133880E 28          0.000000E 00  0.112783E 27         0.000000E 00
     10  0.778935E 28          0.000000E 00 -0.519290E 26         0.000000E 00
     11  0.000000E 00         -0.198094E 21  0.000000E 00         0.507119E 23
ELEMENT             STRESSES
      1 -0.255902E 33 -0.381105E 11
      2 -0.195980E 06  0.241990E 18
      3 -0.195887E 06 -0.290992E 21
      4 -0.195820E 06  0.197302E 03
      5  0.119358E 35 -0.124894E 21
      6 -0.119186E 35 -0.254618E 21
      7  0.196011E 06 -0.109122E 21
      8  0.195954E 06  0.109122E 21
      9  0.195971E 06  0.145496E 21
     10 -0.253560E 23  0.211725E 10
```

A.4.5 Solution of two-dimensional elasto-plastic problem. Example of Section 7.9, Fig. 7.12

Input data

```
2-D ELASTO-PLASTIC EXAMPLE , SECTION 7.9 , FIG 7.12
51   12   18    2    8    1    2    2    2    1    3
 1    1    1    8   12   13   14    9    3    2
 2    1    3    9   14   15   16   10    5    4
 3    1    5   10   16   17   18   11    7    6
 4    1   12   19   23   24   25   20   14   13
 5    1   14   20   25   26   27   21   16   15
 6    1   16   21   27   28   29   22   18   17
 7    1   23   30   34   35   36   31   25   24
 8    1   25   31   36   37   38   32   27   26
 9    1   27   32   38   39   40   33   29   28
10    1   34   41   45   46   47   42   36   35
11    1   36   42   47   48   49   43   38   37
12    1   38   43   49   50   51   44   40   39
 1 100.0        0.0                 27  70.0       121.243
 2  96.592     25.882               28  36.234     135.230
 3  86.602     50.0                 29   0.0       140.0
 4  70.710     70.710               30 155.0         0.0
 5  50.0       86.602               31 134.234      77.5
 6  25.882     96.592               32  77.5       134.234
 7   0.0      100.0                 33   0.0       155.0
 8 110.0        0.0                 34 170.0         0.0
 9  95.263     55.0                 35 164.207      43.999
10  55.0       95.263               36 147.224      85.0
11   0.0      110.0                 37 120.208     120.208
12 120.0        0.0                 38  85.0       147.224
13 115.911     31.058               39  43.999     164.207
14 103.923     60.0                 40   0.0       170.0
15  84.853     84.853               41 185.0         0.0
16  60.0      103.923               42 160.215      92.5
17  31.058    115.911               43  92.5       160.215
18   0.0      120.0                 44   0.0       185.0
19 130.0        0.0                 45 200.0         0.0
20 112.583     65.0                 46 193.185      51.764
21  65.0      112.583               47 173.205     100.0
22   0.0      130.0                 48 141.421     141.421
23 140.0        0.0                 49 100.0       173.205
24 135.230     36.234               50  51.764     193.185
25 121.243     70.0                 51   0.0       200.0
26  98.995     98.995
 1        01        0.0        0.0
 7        10        0.0        0.0
 8        01        0.0        0.0
11        10        0.0        0.0
12        01        0.0        0.0
18        10        0.0        0.0
19        01        0.0        0.0
22        10        0.0        0.0
23        01        0.0        0.0
29        10        0.0        0.0
30        01        0.0        0.0
33        10        0.0        0.0
34        01        0.0        0.0
40        10        0.0        0.0
41        01        0.0        0.0
44        10        0.0        0.0
45        01        0.0        0.0
51        10        0.0        0.0
```

```
  1
21000.0     0.3       0.0       0.0      24.0      0.0       0.0     -0.0
INTERNAL PRESSURE
  0    0    1
  3
  1    3    2    1
 20.0       0.0       20.0      0.0      20.0      0.0
  2    5    4    3
 20.0       0.0       20.0      0.0      20.0      0.0
  3    7    6    5
 20.0       0.0       20.0      0.0      20.0      0.0
 0.7        1.0       30    3    3
```

Line printer output

```
  2-D ELASTO-PLASTIC EXAMPLE , SECTION 7.9 , FIG 7.12
NPOIN =  51     NELEM =  12     NVFIX =  18     NTYPE =   2     NNODE =   8
NMATS =   1     NGAUS =   2     NEVAB =  16     NALGO =   2
NCRIT =   2     NINCS =   1     NSTRE =   3
ELEMENT   PROPERTY       NODE NUMBERS
    1         1           1    8   12   13   14    9    3    2
    2         1           3    9   14   15   16   10    5    4
    3         1           5   10   16   17   18   11    7    6
    4         1          12   19   23   24   25   20   14   13
    5         1          14   20   25   26   27   21   16   15
    6         1          16   21   27   28   29   22   18   17
    7         1          23   30   34   35   36   31   25   24
    8         1          25   31   36   37   38   32   27   26
    9         1          27   32   38   39   40   33   29   28
   10         1          34   41   45   46   47   42   36   35
   11         1          36   42   47   48   49   43   38   37
   12         1          38   43   49   50   51   44   40   39
NODE         X           Y
    1    100.000      0.000                     27     70.000    121.243
    2     96.592     25.882                     28     36.234    135.230
    3     86.602     50.000                     29      0.000    140.000
    4     70.710     70.710                     30    155.000      0.000
    5     50.000     86.602                     31    134.234     77.500
    6     25.882     96.592                     32     77.500    134.234
    7      0.000    100.000                     33      0.000    155.000
    8    110.000      0.000                     34    170.000      0.000
    9     95.263     55.000                     35    164.207     43.999
   10     55.000     95.263                     36    147.224     85.000
   11      0.000    110.000                     37    120.208    120.208
   12    120.000      0.000                     38     85.000    147.224
   13    115.911     31.058                     39     43.999    164.207
   14    103.923     60.000                     40      0.000    170.000
   15     84.853     84.853                     41    185.000      0.000
   16     60.000    103.923                     42    160.215     92.500
   17     31.058    115.911                     43     92.500    160.215
   18      0.000    120.000                     44      0.000    185.000
   19    130.000      0.000                     45    200.000      0.000
   20    112.583     65.000                     46    193.185     51.764
   21     65.000    112.583                     47    173.205    100.000
   22      0.000    130.000                     48    141.421    141.421
   23    140.000      0.000                     49    100.000    173.205
   24    135.230     36.234                     50     51.764    193.185
   25    121.243     70.000                     51      0.000    200.000
   26     98.995     98.995
```

```
NODE      CODE      FIXED VALUES
   1         1       0.000000  0.000000
   7        10       0.000000  0.000000
   8         1       0.000000  0.000000
  11        10       0.000000  0.000000
  12         1       0.000000  0.000000
  18        10       0.000000  0.000000
  19         1       0.000000  0.000000
  22        10       0.000000  0.000000
  23         1       0.000000  0.000000
  29        10       0.000000  0.000000
  30         1       0.000000  0.000000
  33        10       0.000000  0.000000
  34         1       0.000000  0.000000
  40        10       0.000000  0.000000
  41         1       0.000000  0.000000
  44        10       0.000000  0.000000
  45         1       0.000000  0.000000
  51        10       0.000000  0.000000
NUMBER      ELEMENT PROPERTIES
   1    0.210000E 05  0.300000E 00  0.000000E 00  0.000000E 00  0.240000E 02  0.000000E 00  0.000000E 00
MAXIMUM FRONTWIDTH ENCOUNTERED =   24
 INTERNAL PRESSURE
   0    0    1
       NO. OF LOADED EDGES =     3
       LIST OF LOADED EDGES AND APPLIED LOADS
        1          3    2    1
   20.000     0.000     20.000     0.000     20.000     0.000
        2          5    4    3
   20.000     0.000     20.000     0.000     20.000     0.000
        3          7    6    5
   20.000     0.000     20.000     0.000     20.000     0.000
        TOTAL NODAL FORCES FOR EACH ELEMENT
   1      0.1784E 03  0.7800E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.1549E 03  0.8854E 02  0.6667E 03  0.1786E 03
   2      0.1541E 03  0.8989E 02  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.8989E 02  0.1541E 03  0.4880E 03  0.4880E 03
   3      0.8854E 02  0.1549E 03  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.7800E 00  0.1784E 03  0.1786E 03  0.6667E 03
   4      0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
```

	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
5	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
6	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
7	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
8	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
9	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
10	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
11	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
12	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00
	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00	0.0000E 00

INCREMENT NUMBER 1
LOAD FACTOR = 0.70000 CONVERGENCE TOLERANCE = 1.00000 MAX. NO. OF ITERATIONS = 30
INITIAL OUTPUT PARAMETER = 3 FINAL OUTPUT PARAMETER = 3
CONVERGENCE CODE = 1 NORM OF RESIDUAL SUM RATIO = 0.336960E 02 MAXIMUM RESIDUAL = 0.155988E 03
DISPLACEMENTS

NODE	X-DISP.	Y-DISP.			
1	0.127198E 00	0.000000E 00	18	0.000000E 00	0.110185E 00
2	0.122734E 00	0.328877E-01	19	0.103925E 00	0.000000E 00
3	0.110156E 00	0.636002E-01	20	0.900022E-01	0.519632E-01
4	0.898486E-01	0.898486E-01	21	0.519632E-01	0.900022E-01
5	0.636002E-01	0.110156E 00	22	0.000000E 00	0.103925E 00
6	0.328877E-01	0.122734E 00	23	0.987474E-01	0.000000E 00
7	0.000000E 00	0.127198E 00	24	0.953363E-01	0.255449E-01
8	0.117795E 00	0.000000E 00	25	0.855186E-01	0.493745E-01
9	0.102014E 00	0.588984E-01	26	0.697915E-01	0.697915E-01
10	0.588984E-01	0.102014E 00	27	0.493745E-01	0.855186E-01
11	0.000000E 00	0.117795E 00	28	0.255449E-01	0.953363E-01
12	0.110185E 00	0.000000E 00	29	0.000000E 00	0.987474E-01
13	0.106396E 00	0.285087E-01	30	0.924750E-01	0.000000E 00
14	0.954232E-01	0.550931E-01	31	0.800863E-01	0.462379E-01
15	0.778875E-01	0.778875E-01	32	0.462379E-01	0.800863E-01
16	0.550931E-01	0.954232E-01	33	0.000000E 00	0.924750E-01
17	0.285087E-01	0.106396E 00	34	0.876445E-01	0.000000E 00

35	0.846176E-01	0.226732E-01
36	0.759029E-01	0.438226E-01
37	0.619449E-01	0.619449E-01
38	0.438226E-01	0.759029E-01
39	0.226732E-01	0.846176E-01
40	0.000000E 00	0.876445E-01
41	0.838477E-01	0.000000E 00
42	0.726148E-01	0.419240E-01
43	0.419240E-01	0.726148E-01
44	0.000000E 00	0.838477E-01
45	0.808966E-01	0.000000E 00
46	0.781269E-01	0.209340E-01
47	0.700591E-01	0.404483E-01
48	0.571933E-01	0.571933E-01
49	0.404483E-01	0.700591E-01
50	0.209340E-01	0.781269E-01
51	0.000000E 00	0.808966E-01

REACTIONS

NODE	X-REAC.	Y-REAC.
1	0.000000E 00	-0.761999E 02
7	-0.761999E 02	0.000000E 00
8	0.000000E 00	-0.269921E 03
11	-0.269921E 03	0.000000E 00
12	0.000000E 00	-0.116327E 03
18	-0.116327E 03	0.000000E 00
19	0.000000E 00	-0.210260E 03
22	-0.210260E 03	0.000000E 00
23	0.000000E 00	-0.117156E 03
29	-0.117156E 03	0.000000E 00
30	0.000000E 00	-0.250153E 03
33	-0.250153E 03	0.000000E 00
34	0.000000E 00	-0.110283E 03
40	-0.110283E 03	0.000000E 00
41	0.000000E 00	-0.203189E 03
44	-0.203189E 03	0.000000E 00
45	0.000000E 00	-0.465120E 02
51	-0.465120E 02	0.000000E 00

G.P.	XX-STRESS	YY-STRESS	XY-STRESS	ZZ-STRESS	MAX P.S.	MIN P.S.	ANGLE	E.P.S.
ELEMENT NO. =	1							
1	-0.893805E 01	0.180284E 02	-0.307422E 01	0.304329E 01	0.183744E 02	-0.928408E 01	6.422	0.240602E-03
2	-0.485865E 01	0.139487E 02	-0.101400E 02	0.304318E 01	0.183743E 02	-0.928420E 01	23.579	0.240580E-03
3	-0.880961E 01	0.181337E 02	-0.306125E 01	0.280970E 01	0.184771E 02	-0.915305E 01	6.401	0.770100E-05
4	-0.472518E 01	0.140487E 02	-0.101362E 02	0.280953E 01	0.184768E 02	-0.915334E 01	23.599	0.770430E-05
ELEMENT NO. =	2							
1	0.465341E 00	0.862395E 01	-0.132139E 02	0.304290E 01	0.183739E 02	-0.928461E 01	36.422	0.240568E-03
2	0.862395E 01	0.465341E 00	-0.132139E 02	0.304290E 01	0.183739E 02	-0.928461E 01	-36.422	0.240568E-03
3	0.577107E 00	0.874645E 01	-0.131974E 02	0.280952E 01	0.184769E 02	-0.915330E 01	36.401	0.768219E-05
4	0.874645E 01	0.577107E 00	-0.131974E 02	0.280952E 01	0.184769E 02	-0.915330E 01	-36.401	0.768219E-05
ELEMENT NO. =	3							
1	0.139487E 02	-0.485865E 01	-0.101400E 02	0.304318E 01	0.183743E 02	-0.928420E 01	-23.579	0.240580E-03
2	0.180284E 02	-0.893805E 01	-0.307422E 01	0.304329E 01	0.183744E 02	-0.928408E 01	-6.422	0.240602E-03
3	0.140487E 02	-0.472518E 01	-0.101362E 02	0.280953E 01	0.184768E 02	-0.915334E 01	-23.599	0.770431E-05
4	0.181337E 02	-0.880961E 01	-0.306125E 01	0.280970E 01	0.184771E 02	-0.915305E 01	-6.401	0.770100E-05
ELEMENT NO. =	4							
1	-0.713097E 01	0.164644E 02	-0.267828E 01	0.280004E 01	0.167646E 02	-0.743116E 01	6.395	0.000000E 00
2	-0.355180E 01	0.128851E 02	-0.887785E 01	0.280000E 01	0.167646E 02	-0.743124E 01	23.604	0.000000E 00
3	-0.520488E 01	0.145383E 02	-0.224680E 01	0.280002E 01	0.147907E 02	-0.545735E 01	6.411	0.000000E 00

4	-0.221523E 01	0.115483E 02	-0.742551E 01	0.279991E 01	0.147906E 02	-0.545755E 01	23.588	0.000000E 00
ELEMENT NO. =	5							
1	0.108723E 01	0.824570E 01	-0.115562E 02	0.279988E 01	0.167643E 02	-0.743133E 01	36.395	0.000000E 00
2	0.824570E 01	0.108723E 01	-0.115562E 02	0.279988E 01	0.167643E 02	-0.743133E 01	-36.395	0.000000E 00
3	0.167670E 01	0.765648E 01	-0.967249E 01	0.279995E 01	0.147906E 02	-0.545747E 01	36.411	0.000000E 00
4	0.765648E 01	0.167670E 01	-0.967249E 01	0.279995E 01	0.147906E 02	-0.545747E 01	-36.411	0.000000E 00
ELEMENT NO. =	6							
1	0.128851E 02	-0.355180E 01	-0.887785E 01	0.280000E 01	0.167646E 02	-0.743124E 01	-23.604	0.000000E 00
2	0.164644E 02	-0.713097E 01	-0.267828E 01	0.280004E 01	0.167646E 02	-0.743116E 01	-6.395	0.000000E 00
3	0.115483E 02	-0.221523E 01	-0.742551E 01	0.279991E 01	0.147906E 02	-0.545755E 01	-23.588	0.000000E 00
4	0.145383E 02	-0.520488E 01	-0.224680E 01	0.280002E 01	0.147907E 02	-0.545735E 01	-6.411	0.000000E 00
ELEMENT NO. =	7							
1	-0.383616E 01	0.131694E 02	-0.193148E 01	0.279998E 01	0.133861E 02	-0.405278E 01	6.399	0.000000E 00
2	-0.125760E 01	0.105909E 02	-0.639778E 01	0.279999E 01	0.133861E 02	-0.405277E 01	23.600	0.000000E 00
3	-0.212632E 01	0.114596E 02	-0.154577E 01	0.279997E 01	0.116332E 02	-0.229997E 01	6.410	0.000000E 00
4	-0.686952E-01	0.940184E 01	-0.510990E 01	0.279994E 01	0.116332E 02	-0.230005E 01	23.590	0.000000E 00
ELEMENT NO. =	8							
1	0.208787E 01	0.724522E 01	-0.832942E 01	0.279993E 01	0.133860E 02	-0.405291E 01	36.399	0.000000E 00
2	0.724522E 01	0.208787E 01	-0.832942E 01	0.279993E 01	0.133860E 02	-0.405291E 01	-36.399	0.000000E 00
3	0.260888E 01	0.672438E 01	-0.665579E 01	0.279998E 01	0.116333E 02	-0.229999E 01	36.410	0.000000E 00
4	0.672438E 01	0.260888E 01	-0.665579E 01	0.279998E 01	0.116333E 02	-0.229999E 01	-36.410	0.000000E 00
ELEMENT NO. =	9							
1	0.105909E 02	-0.125760E 01	-0.639778E 01	0.279999E 01	0.133861E 02	-0.405277E 01	-23.600	0.000000E 00
2	0.131694E 02	-0.383616E 01	-0.193148E 01	0.279998E 01	0.133861E 02	-0.405278E 01	-6.399	0.000000E 00
3	0.940184E 01	-0.686952E-01	-0.510990E 01	0.279994E 01	0.116332E 02	-0.230005E 01	-23.590	0.000000E 00
4	0.114596E 02	-0.212632E 01	-0.154577E 01	0.279997E 01	0.116332E 02	-0.229997E 01	-6.410	0.000000E 00
ELEMENT NO. =	10							
1	-0.118841E 01	0.105216E 02	-0.132981E 01	0.279995E 01	0.106707E 02	-0.133753E 01	6.398	0.000000E 00
2	0.587478E 00	0.874580E 01	-0.440564E 01	0.279998E 01	0.106707E 02	-0.133746E 01	23.602	0.000000E 00
3	-0.186150E 00	0.951929E 01	-0.110110E 01	0.279994E 01	0.964264E 01	-0.309504E 00	6.392	0.000000E 00
4	0.128661E 01	0.804648E 01	-0.365206E 01	0.279993E 01	0.964263E 01	-0.309548E 00	23.608	0.000000E 00
ELEMENT NO. =	11							
1	0.289070E 01	0.644254E 01	-0.573552E 01	0.279997E 01	0.106708E 02	-0.133755E 01	36.398	0.000000E 00
2	0.644254E 01	0.289070E 01	-0.573552E 01	0.279997E 01	0.106708E 02	-0.133755E 01	-36.398	0.000000E 00
3	0.319390E 01	0.613950E 01	-0.475323E 01	0.280002E 01	0.964288E 01	-0.309476E 00	36.392	0.000000E 00
4	0.613950E 01	0.319390E 01	-0.475323E 01	0.280002E 01	0.964288E 01	-0.309476E 00	-36.392	0.000000E 00
ELEMENT NO. =	12							
1	0.874580E 01	0.587478E 00	-0.440564E 01	0.279998E 01	0.106707E 02	-0.133746E 01	-23.602	0.000000E 00
2	0.105216E 02	-0.118841E 01	-0.132981E 01	0.279995E 01	0.106707E 02	-0.133753E 01	-6.398	0.000000E 00
3	0.804648E 01	0.128661E 01	-0.365206E 01	0.279993E 01	0.964263E 01	-0.309548E 00	-23.608	0.000000E 00

4 0.951929E 01 -0.186150E 00 -0.110110E 01 0.279994E 01 0.964264E 01 -0.309504E 00 -6.392 0.000000E 00

CONVERGENCE CODE = 1 NORM OF RESIDUAL SUM RATIO = 0.118830E 02 MAXIMUM RESIDUAL = 0.416687E 02
CONVERGENCE CODE = 1 NORM OF RESIDUAL SUM RATIO = 0.556571E 01 MAXIMUM RESIDUAL = 0.222848E 02
CONVERGENCE CODE = 1 NORM OF RESIDUAL SUM RATIO = 0.297375E 01 MAXIMUM RESIDUAL = 0.127533E 02
CONVERGENCE CODE = 1 NORM OF RESIDUAL SUM RATIO = 0.165985E 01 MAXIMUM RESIDUAL = 0.728396E 01
CONVERGENCE CODE = 0 NORM OF RESIDUAL SUM RATIO = 0.939223E 00 MAXIMUM RESIDUAL = 0.415713E 01

DISPLACEMENTS

NODE	X-DISP.	Y-DISP.
1	0.139121E 00	0.000000E 00
2	0.134201E 00	0.359609E-01
3	0.120482E 00	0.695626E-01
4	0.982428E-01	0.982428E-01
5	0.695626E-01	0.120482E 00
6	0.359609E-01	0.134201E 00
7	0.000000E 00	0.139121E 00
8	0.127126E 00	0.000000E 00
9	0.110094E 00	0.635643E-01
10	0.635643E-01	0.110094E 00
11	0.000000E 00	0.127126E 00
12	0.118379E 00	0.000000E 00
13	0.114299E 00	0.306268E-01
14	0.102520E 00	0.591908E-01
15	0.836738E-01	0.836738E-01
16	0.591908E-01	0.102520E 00
17	0.306268E-01	0.114299E 00
18	0.000000E 00	0.118379E 00
19	0.111650E 00	0.000000E 00
20	0.966928E-01	0.558264E-01
21	0.558264E-01	0.966928E-01
22	0.000000E 00	0.111650E 00
23	0.106084E 00	0.000000E 00
24	0.102421E 00	0.274436E-01
25	0.918730E-01	0.530435E-01
26	0.749788E-01	0.749788E-01
27	0.530435E-01	0.918730E-01
28	0.274436E-01	0.102421E 00
29	0.000000E 00	0.106084E 00
30	0.993465E-01	0.000000E 00
31	0.860377E-01	0.496741E-01
32	0.496741E-01	0.860377E-01
37	0.665485E-01	0.665485E-01
38	0.470796E-01	0.815441E-01
39	0.243581E-01	0.909056E-01
40	0.000000E 00	0.941578E-01
41	0.900786E-01	0.000000E 00
42	0.780114E-01	0.450397E-01
43	0.450397E-01	0.780114E-01
44	0.000000E 00	0.900786E-01
45	0.869080E-01	0.000000E 00
46	0.839328E-01	0.224896E-01
47	0.752657E-01	0.434542E-01
48	0.614439E-01	0.614439E-01
49	0.434542E-01	0.752657E-01
50	0.224896E-01	0.839328E-01
51	0.000000E 00	0.869080E-01

REACTIONS

NODE	X-REAC.	Y-REAC.
1	0.000000E 00	-0.464276E 02
7	-0.464276E 02	0.000000E 00
8	0.000000E 00	-0.220459E 03
11	-0.220459E 03	0.000000E 00
12	0.000000E 00	-0.125854E 03
18	-0.125854E 03	0.000000E 00
19	0.000000E 00	-0.225928E 03
22	-0.225928E 03	0.000000E 00
23	0.000000E 00	-0.125859E 03
29	-0.125859E 03	0.000000E 00
30	0.000000E 00	-0.268735E 03
33	-0.268735E 03	0.000000E 00
34	0.000000E 00	-0.118479E 03
40	-0.118479E 03	0.000000E 00
41	0.000000E 00	-0.218290E 03

33	0.000000E 00	0.993465E-01
34	0.941578E-01	0.000000E 00
35	0.909056E-01	0.243581E-01
36	0.815441E-01	0.470796E-01
44	-0.218290E 03	0.000000E 00
45	0.000000E 00	-0.499673E 02
51	-0.499673E 02	0.000000E 00

G.P.	XX-STRESS	YY-STRESS	XY-STRESS	ZZ-STRESS	MAX P.S.	MIN P.S.	ANGLE	E.P.S.
ELEMENT NO. =	1							
1	-0.123717E 02	0.146473E 02	-0.308107E 01	0.117112E 01	0.149941E 02	-0.127186E 02	6.424	0.451304E-03
2	-0.828491E 01	0.105605E 02	-0.101593E 02	0.117110E 01	0.149942E 02	-0.127186E 02	23.577	0.451255E-03
3	-0.948121E 01	0.174939E 02	-0.306568E 01	0.257060E 01	0.178380E 02	-0.982523E 01	6.403	0.108534E-03
4	-0.539247E 01	0.134047E 02	-0.101479E 02	0.257044E 01	0.178377E 02	-0.982547E 01	23.598	0.108528E-03
ELEMENT NO. =	2							
1	-0.294888E 01	0.522409E 01	-0.132401E 02	0.117090E 01	0.149940E 02	-0.127188E 02	36.424	0.451200E-03
2	0.522409E 01	-0.294888E 01	-0.132401E 02	0.117090E 01	0.149940E 02	-0.127188E 02	-36.424	0.451200E-03
3	-0.825393E-01	0.809511E 01	-0.132134E 02	0.257046E 01	0.178379E 02	-0.982530E 01	36.403	0.108473E-03
4	0.809511E 01	-0.825394E-01	-0.132134E 02	0.257046E 01	0.178379E 02	-0.982530E 01	-36.403	0.108473E-03
ELEMENT NO. =	3							
1	0.105605E 02	-0.828491E 01	-0.101593E 02	0.117110E 01	0.149942E 02	-0.127186E 02	-23.577	0.451255E-03
2	0.146473E 02	-0.123717E 02	-0.308107E 01	0.117112E 01	0.149941E 02	-0.127186E 02	-6.424	0.451304E-03
3	0.134047E 02	-0.539247E 01	-0.101479E 02	0.257044E 01	0.178377E 02	-0.982547E 01	-23.598	0.108528E-03
4	0.174939E 02	-0.948121E 01	-0.306568E 01	0.257060E 01	0.178380E 02	-0.982523E 01	-6.403	0.108534E-03
ELEMENT NO. =	4							
1	-0.766058E 01	0.176878E 02	-0.287878E 01	0.300817E 01	0.180106E 02	-0.798341E 01	6.398	0.000000E 00
2	-0.381672E 01	0.138438E 02	-0.953667E 01	0.300813E 01	0.180105E 02	-0.798344E 01	23.601	0.000000E 00
3	-0.559170E 01	0.156189E 02	-0.241350E 01	0.300815E 01	0.158900E 02	-0.586286E 01	6.410	0.000000E 00
4	-0.237967E 01	0.124063E 02	-0.797755E 01	0.300798E 01	0.158898E 02	-0.586315E 01	23.589	0.000000E 00
ELEMENT NO. =	5							
1	0.116933E 01	0.885683E 01	-0.124153E 02	0.300785E 01	0.180098E 02	-0.798366E 01	36.399	0.000000E 00
2	0.885683E 01	0.116933E 01	-0.124153E 02	0.300785E 01	0.180098E 02	-0.798366E 01	-36.399	0.000000E 00
3	0.180098E 01	0.822568E 01	-0.103912E 02	0.300800E 01	0.158897E 02	-0.586305E 01	36.411	0.000000E 00
4	0.822568E 01	0.180098E 01	-0.103912E 02	0.300800E 01	0.158897E 02	-0.586305E 01	-36.411	0.000000E 00
ELEMENT NO. =	6							
1	0.138438E 02	-0.381672E 01	-0.953667E 01	0.300813E 01	0.180105E 02	-0.798344E 01	-23.601	0.000000E 00
2	0.176878E 02	-0.766058E 01	-0.287878E 01	0.300817E 01	0.180106E 02	-0.798341E 01	-6.398	0.000000E 00
3	0.124063E 02	-0.237967E 01	-0.797755E 01	0.300798E 01	0.158898E 02	-0.586315E 01	-23.589	0.000000E 00
4	0.156189E 02	-0.559170E 01	-0.241350E 01	0.300815E 01	0.158900E 02	-0.586286E 01	-6.410	0.000000E 00
ELEMENT NO. =	7							
1	-0.412127E 01	0.141482E 02	-0.207478E 01	0.300809E 01	0.143809E 02	-0.435393E 01	6.398	0.000000E 00
2	-0.135088E 01	0.113778E 02	-0.687337E 01	0.300808E 01	0.143809E 02	-0.435393E 01	23.601	0.000000E 00
3	-0.228431E 01	0.123112E 02	-0.166069E 01	0.300806E 01	0.124977E 02	-0.247088E 01	6.410	0.000000E 00
4	-0.738630E-01	0.101006E 02	-0.548958E 01	0.300802E 01	0.124977E 02	-0.247098E 01	23.589	0.000000E 00

```
  ELEMENT NO. =     8
 1    0.224272E 01  0.778385E 01 -0.894834E 01  0.300797E 01  0.143807E 02 -0.435415E 01  36.398  0.000000E 00
 2    0.778385E 01  0.224272E 01 -0.894834E 01  0.300797E 01  0.143807E 02 -0.435415E 01 -36.398  0.000000E 00
 3    0.280277E 01  0.722406E 01 -0.715043E 01  0.300805E 01  0.124978E 02 -0.247095E 01  36.410  0.000000E 00
 4    0.722406E 01  0.280277E 01 -0.715043E 01  0.300805E 01  0.124978E 02 -0.247095E 01 -36.410  0.000000E 00
  ELEMENT NO. =     9
 1    0.113778E 02 -0.135088E 01 -0.687337E 01  0.300808E 01  0.143809E 02 -0.435393E 01 -23.601  0.000000E 00
 2    0.141482E 02 -0.412127E 01 -0.207478E 01  0.300809E 01  0.143809E 02 -0.435393E 01  -6.398  0.000000E 00
 3    0.101006E 02 -0.738630E-01 -0.548958E 01  0.300802E 01  0.124977E 02 -0.247098E 01 -23.589  0.000000E 00
 4    0.123112E 02 -0.228431E 01 -0.166069E 01  0.300806E 01  0.124977E 02 -0.247088E 01  -6.410  0.000000E 00
  ELEMENT NO. =    10
 1   -0.127671E 01  0.113035E 02 -0.142867E 01  0.300803E 01  0.114637E 02 -0.143691E 01   6.398  0.000000E 00
 2    0.631079E 00  0.939580E 01 -0.473299E 01  0.300806E 01  0.114637E 02 -0.143686E 01  23.601  0.000000E 00
 3   -0.199987E 00  0.102267E 02 -0.118290E 01  0.300800E 01  0.103592E 02 -0.332502E 00   6.392  0.000000E 00
 4    0.138223E 01  0.864445E 01 -0.392346E 01  0.300800E 01  0.103592E 02 -0.332548E 00  23.608  0.000000E 00
  ELEMENT NO. =    11
 1    0.310555E 01  0.692130E 01 -0.616178E 01  0.300805E 01  0.114638E 02 -0.143697E 01  36.398  0.000000E 00
 2    0.692130E 01  0.310555E 01 -0.616178E 01  0.300805E 01  0.114638E 02 -0.143697E 01 -36.398  0.000000E 00
 3    0.343125E 01  0.659581E 01 -0.510649E 01  0.300812E 01  0.103595E 02 -0.332480E 00  36.392  0.000000E 00
 4    0.659581E 01  0.343125E 01 -0.510649E 01  0.300812E 01  0.103595E 02 -0.332480E 00 -36.392  0.000000E 00
  ELEMENT NO. =    12
 1    0.939580E 01  0.631079E 00 -0.473299E 01  0.300806E 01  0.114637E 02 -0.143686E 01 -23.601  0.000000E 00
 2    0.113035E 02 -0.127671E 01 -0.142867E 01  0.300803E 01  0.114637E 02 -0.143691E 01  -6.398  0.000000E 00
 3    0.864445E 01  0.138223E 01 -0.392346E 01  0.300800E 01  0.103592E 02 -0.332548E 00 -23.608  0.000000E 00
 4    0.102267E 02 -0.199987E 00 -0.118290E 01  0.300800E 01  0.103592E 02 -0.332502E 00  -6.392  0.000000E 00
```

A.4.6 Solution of two-dimensional elasto-viscoplastic problem. Example of Section 8.16, Fig. 8.10

Input data

```
  2-D ELASTO - VISCOPLASTIC EXAMPLE , SECTION 8.16 , FIG. 8.10
51   12   18    2    8    1    2    2    2    1    3
 1    1    1    8   12   13   14    9    3    2
 2    1    3    9   14   15   16   10    5    4
 3    1    5   10   16   17   18   11    7    6
 4    1   12   19   23   24   25   20   14   13
```

5	1	14	20	25	26	27	21	16	15
6	1	16	21	27	28	29	22	18	17
7	1	23	30	34	35	36	31	25	24
8	1	25	31	36	37	38	32	27	26
9	1	27	32	38	39	40	33	29	28
10	1	34	41	45	46	47	42	36	35
11	1	36	42	47	48	49	43	38	37
12	1	38	43	49	50	51	44	40	39

1	100.0	0.0
2	96.592	25.882
3	86.602	50.0
4	70.710	70.710
5	50.0	86.602
6	25.882	96.592
7	0.0	100.0
8	110.0	0.0
9	95.263	55.0
10	55.0	95.263
11	0.0	110.0
12	120.0	0.0
13	115.911	31.058
14	103.923	60.0
15	84.853	84.853
16	60.0	103.923
17	31.058	115.911
18	0.0	120.0
19	130.0	0.0
20	112.583	65.0
21	65.0	112.583
22	0.0	130.0
23	140.0	0.0
24	135.230	36.234
25	121.243	70.0
26	98.995	98.995
27	70.0	121.243
28	36.234	135.230
29	0.0	140.0
30	155.0	0.0
31	134.234	77.5
32	77.5	134.234
33	0.0	155.0
34	170.0	0.0
35	164.207	43.999
36	147.224	85.0
37	120.208	120.208
38	85.0	147.224
39	43.999	164.207
40	0.0	170.0
41	185.0	0.0
42	160.215	92.5
43	92.5	160.215
44	0.0	185.0
45	200.0	0.0
46	193.185	51.764
47	173.205	100.0
48	141.421	141.421
49	100.0	173.205
50	51.764	193.185
51	0.0	200.0

1	01	0.0	0.0
7	10	0.0	0.0
8	01	0.0	0.0
11	10	0.0	0.0
12	01	0.0	0.0
18	10	0.0	0.0

```
  19        01      0.0      0.0
  22        10      0.0      0.0
  23        01      0.0      0.0
  29        10      0.0      0.0
  30        01      0.0      0.0
  33        10      0.0      0.0
  34        01      0.0      0.0
  40        10      0.0      0.0
  41        01      0.0      0.0
  44        10      0.0      0.0
  45        01      0.0      0.0
  51        10      0.0      0.0
   1
21000.0      0.3      0.0      0.0     24.0      0.0      0.0      0.001
  1.0       1.0
INTERNAL PRESSURE
   0    0    1
   3
   1    3    2    1
  20.0      0.0     20.0      0.0     20.0      0.0
   2    5    4    3
  20.0      0.0     20.0      0.0     20.0      0.0
   3    7    6    5
  20.0      0.0     20.0      0.0     20.0      0.0
   0.0      0.05     0.1      1.5
  0.7      0.1      50   10   10
```

Line printer output

```
     2-D ELASTO - VISCOPLASTIC EXAMPLE , SECTION 8.16 , FIG. 8.10
NPOIN =  51     NELEM =  12     NVFIX =  18     NTYPE =   2     NNODE =   8
NMATS =   1     NGAUS =   2     NEVAB =  16     NALGO =   2
NCRIT =   2     NINCS =   1     NSTRE =   3
ELEMENT    PROPERTY       NODE NUMBERS
    1         1            1    8   12   13   14    9    3    2
    2         1            3    9   14   15   16   10    5    4
    3         1            5   10   16   17   18   11    7    6
    4         1           12   19   23   24   25   20   14   13
    5         1           14   20   25   26   27   21   16   15
    6         1           16   21   27   28   29   22   18   17
```

7	1	23	30	34	35	36	31	25	24
8	1	25	31	36	37	38	32	27	26
9	1	27	32	38	39	40	33	29	28
10	1	34	41	45	46	47	42	36	35
11	1	36	42	47	48	49	43	38	37
12	1	38	43	49	50	51	44	40	39

NODE	X	Y
1	100.000	0.000
2	96.592	25.882
3	86.602	50.000
4	70.710	70.710
5	50.000	86.602
6	25.882	96.592
7	0.000	100.000
8	110.000	0.000
9	95.263	55.000
10	55.000	95.263
11	0.000	110.000
12	120.000	0.000
13	115.911	31.058
14	103.923	60.000
15	84.853	84.853
16	60.000	103.923
17	31.058	115.911
18	0.000	120.000
19	130.000	0.000
20	112.583	65.000
21	65.000	112.583
22	0.000	130.000
23	140.000	0.000
24	135.230	36.234
25	121.243	70.000
26	98.995	98.995
27	70.000	121.243
28	36.234	135.230
29	0.000	140.000
30	155.000	0.000
31	134.234	77.500
32	77.500	134.234
33	0.000	155.000
34	170.000	0.000
35	164.207	43.999
36	147.224	85.000
37	120.208	120.208
38	85.000	147.224
39	43.999	164.207
40	0.000	170.000
41	185.000	0.000
42	160.215	92.500
43	92.500	160.215
44	0.000	185.000
45	200.000	0.000
46	193.185	51.764
47	173.205	100.000
48	141.421	141.421
49	100.000	173.205
50	51.764	193.185
51	0.000	200.000

NODE	CODE	FIXED VALUES	
1	1	0.000000	0.000000
7	10	0.000000	0.000000
8	1	0.000000	0.000000
11	10	0.000000	0.000000
12	1	0.000000	0.000000
18	10	0.000000	0.000000

```
  19         1       0.000000  0.000000
  22        10       0.000000  0.000000
  23         1       0.000000  0.000000
  29        10       0.000000  0.000000
  30         1       0.000000  0.000000
  33        10       0.000000  0.000000
  34         1       0.000000  0.000000
  40        10       0.000000  0.000000
  41         1       0.000000  0.000000
  44        10       0.000000  0.000000
  45         1       0.000000  0.000000
  51        10       0.000000  0.000000
NUMBER      ELEMENT PROPERTIES
   1     0.210000E 05  0.300000E 00  0.000000E 00  0.000000E 00  0.240000E 02  0.000000E 00  0.000000E 00  0.100000E-02
         0.100000E 01  0.100000E 01
MAXIMUM FRONTWIDTH ENCOUNTERED =   24
 INTERNAL PRESSURE
   0    0    1
     NO. OF LOADED EDGES =    3
     LIST OF LOADED EDGES AND APPLIED LOADS
         1          3    2    1
  20.000     0.000     20.000      0.000     20.000      0.000
         2          5    4    3
  20.000     0.000     20.000      0.000     20.000      0.000
         3          7    6    5
  20.000     0.000     20.000      0.000     20.000      0.000
      TOTAL NODAL FORCES FOR EACH ELEMENT
   1       0.1784E 03  0.7800E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
           0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.1549E 03  0.8854E 02  0.6667E 03  0.1786E 03
   2       0.1541E 03  0.8989E 02  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
           0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.8989E 02  0.1541E 03  0.4880E 03  0.4880E 03
   3       0.8854E 02  0.1549E 03  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
           0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.7800E 00  0.1784E 03  0.1786E 03  0.6667E 03
   4       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
           0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
   5       0.0000E 00  0.0000E 00  0.000CE 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
           0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
   6       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
           0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
   7       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
```

```
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
  8       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
  9       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
 10       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
 11       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
 12       0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
          0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00  0.0000E 00
   TIME STEPPING PARAMETER =      0.000     TIME STEP STABILITY FACTOR =   0.05000
  INITIAL TIME STEP LENGTH =    0.10000     TIME STEP INCREMENT PARAMETER =    1.50000
   INCREMENT NUMBER      1
   LOAD FACTOR =    0.70000      CONVERGENCE TOLERANCE =   0.10000     MAX. NO. OF ITERATIONS =   50
INITIAL OUTPUT PARAMETER =   10     FINAL OUTPUT PARAMETER =   10
   TOTAL TIME =      0.000000E 00
 CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.100000E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.100000E 00
 CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.148250E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.250000E 00
 CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.207778E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.475000E 00
 CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.280997E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.812500E 00
 CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.313019E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.125353E 01
 CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.340506E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.184786E 01
 CONVERGENCE CODE = 999   NORM OF RESIDUAL SUM RATIO =  0.377261E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.273772E 01
 CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.345160E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.407250E 01
 CONVERGENCE CODE =   1   NORM OF RESIDUAL SUM RATIO =  0.213414E 03   MAXIMUM RESIDUAL =  0.000000E 00
   TOTAL TIME =      0.607467E 01
 CONVERGENCE CODE =   0   NORM OF RESIDUAL SUM RATIO =  0.000000E 00   MAXIMUM RESIDUAL =  0.000000E 00
   DISPLACEMENTS
    NODE      X-DISP.       Y-DISP.
      1  0.139590E 00  0.000000E 00                      3  0.120888E 00  0.697974E-01
      2  0.134655E 00  0.360826E-01                      4  0.985748E-01  0.985748E-01
```

5	0.697974E-01	0.120888E 00
6	0.360826E-01	0.134655E 00
7	0.000000E 00	0.139590E 00
8	0.127595E 00	0.000000E 00
9	0.110501E 00	0.637993E-01
10	0.637993E-01	0.110501E 00
11	0.000000E 00	0.127595E 00
12	0.118811E 00	0.000000E 00
13	0.114717E 00	0.307387E-01
14	0.102894E 00	0.594071E-01
15	0.839794E-01	0.839794E-01
16	0.594071E-01	0.102894E 00
17	0.307387E-01	0.114717E 00
18	0.000000E 00	0.118811E 00
19	0.112058E 00	0.000000E 00
20	0.970459E-01	0.560303E-01
21	0.560303E-01	0.970459E-01
22	0.000000E 00	0.112058E 00
23	0.106472E 00	0.000000E 00
24	0.102796E 00	0.275438E-01
25	0.922085E-01	0.532372E-01
26	0.752527E-01	0.752527E-01
27	0.532372E-01	0.922085E-01
28	0.275438E-01	0.102796E 00
29	0.000000E 00	0.106472E 00
30	0.997092E-01	0.000000E 00
31	0.863519E-01	0.498555E-01
32	0.498555E-01	0.863519E-01
33	0.000000E 00	0.997092E-01
34	0.945016E-01	0.000000E 00
35	0.912376E-01	0.244471E-01
36	0.818419E-01	0.472516E-01
37	0.667916E-01	0.667916E-01
38	0.472516E-01	0.818419E-01
39	0.244471E-01	0.912376E-01
40	0.000000E 00	0.945016E-01
41	0.904075E-01	0.000000E 00
42	0.782963E-01	0.452042E-01
43	0.452042E-01	0.782963E-01
44	0.000000E 00	0.904075E-01
45	0.872253E-01	0.000000E 00
46	0.842393E-01	0.225717E-01
47	0.755406E-01	0.436128E-01
48	0.616684E-01	0.616684E-01
49	0.436128E-01	0.755406E-01
50	0.225717E-01	0.842393E-01
51	0.000000E 00	0.872253E-01

REACTIONS

NODE	X-REAC.	Y-REAC.
1	0.000000E 00	-0.456968E 02
7	-0.456968E 02	0.000000E 00
8	0.000000E 00	-0.217851E 03
11	-0.217851E 03	0.000000E 00
12	0.000000E 00	-0.125513E 03
18	-0.125513E 03	0.000000E 00
19	0.000000E 00	-0.226754E 03
22	-0.226754E 03	0.000000E 00
23	0.000000E 00	-0.126319E 03
29	-0.126319E 03	0.000000E 00
30	0.000000E 00	-0.269717E 03
33	-0.269717E 03	0.000000E 00
34	0.000000E 00	-0.118912E 03
40	-0.118912E 03	0.000000E 00
41	0.000000E 00	-0.219087E 03
44	-0.219087E 03	0.000000E 00
45	0.000000E 00	-0.501497E 02
51	-0.501497E 02	0.000000E 00

G.P.	XX-STRESS	YY-STRESS	XY-STRESS	ZZ-STRESS	MAX P.S.	MIN P.S.	ANGLE	E.P.S.
ELEMENT NO. =	1							
1	-0.125015E 02	0.145585E 02	-0.308575E 01	0.617103E 00	0.149059E 02	-0.128489E 02	6.424	0.452901E-03
2	-0.840843E 01	0.104656E 02	-0.101747E 02	0.617146E 00	0.149060E 02	-0.128488E 02	23.577	0.452852E-03
3	-0.959430E 01	0.174053E 02	-0.306854E 01	0.234329E 01	0.177496E 02	-0.993866E 01	6.403	0.112244E-03
4	-0.550191E 01	0.133124E 02	-0.101570E 02	0.234314E 01	0.177494E 02	-0.993889E 01	23.598	0.112237E-03

ELEMENT NO. =	2							
1	-0.306428E 01	0.512105E 01	-0.132601E 02	0.617031E 00	0.149057E 02	-0.128490E 02	36.424	0.452796E-03
2	0.512105E 01	-0.306428E 01	-0.132601E 02	0.617031E 00	0.149057E 02	-0.128490E 02	-36.424	0.452796E-03
3	-0.187011E 00	0.799786E 01	-0.132254E 02	0.234325E 01	0.177495E 02	-0.993866E 01	36.403	0.112182E-03
4	0.799786E 01	-0.187011E 00	-0.132254E 02	0.234325E 01	0.177495E 02	-0.993866E 01	-36.403	0.112182E-03
ELEMENT NO. =	3							
1	0.104656E 02	-0.840843E 01	-0.101747E 02	0.617146E 00	0.149060E 02	-0.128488E 02	-23.577	0.452852E-03
2	0.145585E 02	-0.125015E 02	-0.308575E 01	0.617103E 00	0.149059E 02	-0.128489E 02	-6.424	0.452901E-03
3	0.133124E 02	-0.550191E 01	-0.101570E 02	0.234314E 01	0.177494E 02	-0.993889E 01	-23.598	0.112237E-03
4	0.174053E 02	-0.959430E 01	-0.306854E 01	0.234329E 01	0.177496E 02	-0.993866E 01	-6.403	0.112244E-03
ELEMENT NO. =	4							
1	-0.768855E 01	0.177524E 02	-0.288931E 01	0.301916E 01	0.180764E 02	-0.801257E 01	6.398	0.000000E 00
2	-0.383066E 01	0.138944E 02	-0.957149E 01	0.301912E 01	0.180763E 02	-0.801259E 01	23.601	0.000000E 00
3	-0.561211E 01	0.156759E 02	-0.242231E 01	0.301914E 01	0.159481E 02	-0.588426E 01	6.410	0.000000E 00
4	-0.238836E 01	0.124516E 02	-0.800669E 01	0.301897E 01	0.159478E 02	-0.588457E 01	23.589	0.000000E 00
ELEMENT NO. =	5							
1	0.117360E 01	0.888913E 01	-0.124607E 02	0.301882E 01	0.180756E 02	-0.801283E 01	36.399	0.000000E 00
2	0.888913E 01	0.117360E 01	-0.124607E 02	0.301882E 01	0.180756E 02	-0.801283E 01	-36.399	0.000000E 00
3	0.180755E 01	0.825572E 01	-0.104291E 02	0.301898E 01	0.159477E 02	-0.588446E 01	36.411	0.000000E 00
4	0.825572E 01	0.180755E 01	-0.104291E 02	0.301898E 01	0.159477E 02	-0.588446E 01	-36.411	0.000000E 00
ELEMENT NO. =	6							
1	0.138944E 02	-0.383066E 01	-0.957149E 01	0.301912E 01	0.180763E 02	-0.801259E 01	-23.601	0.000000E 00
2	0.177524E 02	-0.768855E 01	-0.288931E 01	0.301916E 01	0.180764E 02	-0.801257E 01	-6.398	0.000000E 00
3	0.124516E 02	-0.238836E 01	-0.800669E 01	0.301897E 01	0.159478E 02	-0.588457E 01	-23.589	0.000000E 00
4	0.156759E 02	-0.561211E 01	-0.242231E 01	0.301914E 01	0.159481E 02	-0.588426E 01	-6.410	0.000000E 00
ELEMENT NO. =	7							
1	-0.413632E 01	0.141999E 02	-0.208235E 01	0.301908E 01	0.144334E 02	-0.436983E 01	6.398	0.000000E 00
2	-0.135581E 01	0.114194E 02	-0.689847E 01	0.301907E 01	0.144334E 02	-0.436983E 01	23.601	0.000000E 00
3	-0.229264E 01	0.123561E 02	-0.166675E 01	0.301905E 01	0.125434E 02	-0.247990E 01	6.410	0.000000E 00
4	-0.741370E-01	0.101375E 02	-0.550962E 01	0.301901E 01	0.125434E 02	-0.248000E 01	23.589	0.000000E 00
ELEMENT NO. =	8							
1	0.225090E 01	0.781227E 01	-0.898102E 01	0.301895E 01	0.144332E 02	-0.437006E 01	36.398	0.000000E 00
2	0.781227E 01	0.225090E 01	-0.898102E 01	0.301895E 01	0.144332E 02	-0.437006E 01	-36.398	0.000000E 00
3	0.281300E 01	0.725044E 01	-0.717655E 01	0.301903E 01	0.125434E 02	-0.247998E 01	36.410	0.000000E 00
4	0.725044E 01	0.281300E 01	-0.717655E 01	0.301903E 01	0.125434E 02	-0.247998E 01	-36.410	0.000000E 00
ELEMENT NO. =	9							
1	0.114194E 02	-0.135581E 01	-0.689847E 01	0.301907E 01	0.144334E 02	-0.436983E 01	-23.601	0.000000E 00
2	0.141999E 02	-0.413632E 01	-0.208235E 01	0.301908E 01	0.144334E 02	-0.436983E 01	-6.398	0.000000E 00
3	0.101375E 02	-0.741370E-01	-0.550962E 01	0.301901E 01	0.125434E 02	-0.248000E 01	-23.589	0.000000E 00
4	0.123561E 02	-0.229264E 01	-0.166675E 01	0.301905E 01	0.125434E 02	-0.247990E 01	-6.410	0.000000E 00

```
    ELEMENT NO. =    10
 1    -0.128137E 01  0.113448E 02 -0.143389E 01  0.301902E 01  0.115056E 02 -0.144216E 01   6.398  0.000000E 00
 2     0.633380E 00  0.943011E 01 -0.475028E 01  0.301905E 01  0.115056E 02 -0.144210E 01  23.601  0.000000E 00
 3    -0.200717E 00  0.102640E 02 -0.118721E 01  0.301899E 01  0.103970E 02 -0.333716E 00   6.392  0.000000E 00
 4     0.138728E 01  0.867602E 01 -0.393779E 01  0.301899E 01  0.103971E 02 -0.333762E 00  23.608  0.000000E 00
    ELEMENT NO. =    11
 1     0.311689E 01  0.694658E 01 -0.618428E 01  0.301904E 01  0.115057E 02 -0.144222E 01  36.398  0.000000E 00
 2     0.694658E 01  0.311689E 01 -0.618428E 01  0.301904E 01  0.115057E 02 -0.144222E 01 -36.398  0.000000E 00
 3     0.344379E 01  0.661991E 01 -0.512514E 01  0.301911E 01  0.103974E 02 -0.333695E 00  36.392  0.000000E 00
 4     0.661991E 01  0.344379E 01 -0.512514E 01  0.301911E 01  0.103974E 02 -0.333695E 00 -36.392  0.000000E 00
    ELEMENT NO. =    12
 1     0.943011E 01  0.633380E 00 -0.475028E 01  0.301905E 01  0.115056E 02 -0.144210E 01 -23.601  0.000000E 00
 2     0.113448E 02 -0.128137E 01 -0.143389E 01  0.301902E 01  0.115056E 02 -0.144216E 01  -6.398  0.000000E 00
 3     0.867602E 01  0.138728E 01 -0.393779E 01  0.301899E 01  0.103971E 02 -0.333762E 00 -23.608  0.000000E 00
 4     0.102640E 02 -0.200717E 00 -0.118721E 01  0.301899E 01  0.103970E 02 -0.333716E 00  -6.392  0.000000E 00
```

A.4.7 Solution of a non-layered elasto-plastic Mindlin plate. Example of Section 9.7, Fig. 9.6

Input data

```
MINDLIN NON-LAYERED EXAMPLE,  SECTION 9.7,  FIG. 9.6
   25    4    1    5    9    1    3    2    1   39    0
    1    1    1    2    3    8   13   12   11    6    7
    2    1    3    4    5   10   15   14   13    8    9
    3    1   11   12   13   18   23   22   21   16   17
    4    1   13   14   15   20   25   24   23   18   19
    1       0.0        0.0
    3       0.25       0.0
    5       0.5        0.0
   11       0.0        0.25
   13       0.25       0.25
   15       0.50       0.25
   21       0.0        0.50
   23       0.25       0.50
   25       0.50       0.50
    1  111
    2  110
```

```
    3  110
    4  110
    5  110
    6  101
   10  010
   11  101
   15  010
   16  101
   20  010
   21  101
   22  001
   23  001
   24  001
   25  011
    1
     10.92       0.3      0.01       1.0                 0.04
1111
1111
 UNIFORMLY DISTRIBUTED LOADING INTENSITY -0.01LB/SQ INCH
    0
       0.5       0.1   60    0    3          0.005     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.02       0.1   60    0    3          0.002     0.1   60    0    3
      0.01       0.1   60    0    3          0.002     0.1   60    0    3
      0.01       0.1   60    0    3          0.002     0.1   60    0    3
     0.005       0.1   60    0    3          0.002     0.1   60    0    3
     0.005       0.1   60    0    3          0.002     0.1   60    0    3
     0.005       0.1   60    0    3          0.002     0.1   60    0    3
```

Line printer output

```
MINDLIN NON-LAYERED EXAMPLE,  SECTION 9.7,  FIG. 9.6
NPOIN =  25     NELEM =   4     NVFIX =  16     NTYPE =   5     NNODE =  9
NMATS =   1     NGAUS =   3     NEVAB =  27     NALGO =   2
NCRIT =   1     NINCS =  39     NLAPS =   0     NSWIT =   0
ELEMENT   PROPERTY       NODE NUMBERS
    1         1           1    2    3    8   13   12   11    6    7
    2         1           3    4    5   10   15   14   13    8    9
    3         1          11   12   13   18   23   22   21   16   17
    4         1          13   14   15   20   25   24   23   18   19
NODE        X          Y
   1     0.00000    0.00000
   2      .12500    0.00000
   3      .25000    0.00000
   4      .37500    0.00000
   5      .50000    0.00000
   6     0.00000     .12500
   7     0.00000    0.00000
   8      .25000     .12500
   9     0.00000    0.00000
  10      .50000     .12500
  11     0.00000     .25000
  12      .12500     .25000
  13      .25000     .25000
  14      .37500     .25000
  15      .50000     .25000
  16     0.00000     .37500
  17     0.00000    0.00000
  18      .25000     .37500
  19     0.00000    0.00000
  20      .50000     .37500
  21     0.00000     .50000
  22      .12500     .50000
  23      .25000     .50000
  24      .37500     .50000
  25      .50000     .50000
NODE      CODE       FIXED VALUES
   1       111         0.000000  0.000000  0.000000
   2       110         0.000000  0.000000  0.000000
   3       110         0.000000  0.000000  0.000000
```

4	110	0.000000	0.000000	0.000000
5	110	0.000000	0.000000	0.000000
6	101	0.000000	0.000000	0.000000
10	10	0.000000	0.000000	0.000000
11	101	0.000000	0.000000	0.000000
15	10	0.000000	0.000000	0.000000
16	101	0.000000	0.000000	0.000000
20	10	0.000000	0.000000	0.000000
21	101	0.000000	0.000000	0.000000
22	1	0.000000	0.000000	0.000000
23	1	0.000000	0.000000	0.000000
24	1	0.000000	0.000000	0.000000
25	11	0.000000	0.000000	0.000000

NUMBER ELEMENT PROPERTIES

3	.4472	.0528	.14846E-01	.14812E-01	-.67909E-02	0.
4	.4472	.1972	.20658E-02	.21696E-02	-.51738E-02	0.
5	.3750	.1250	.93182E-02	.10703E-01	-.45473E-02	0.
6	.3750	.2218	.16282E-01	.19151E-01	-.37263E-02	0.
7	.4718	.0282	.32303E-02	.51552E-02	-.17322E-02	0.
8	.4718	.1250	.10243E-01	.11671E-01	-.53073E-03	0.
9	.4718	.2218	.16015E-01	.17815E-01	-.10205E-02	0.

ELEMENT NO. = 3

1	.0528	.3028	.42854E-02	.24316E-02	-.83085E-02	0.
2	.0528	.4472	.21696E-02	.20658E-02	-.51738E-02	0.
3	.1972	.3028	.51552E-02	.32303E-02	-.17322E-02	0.
4	.1972	.4472	.94492E-02	.83078E-02	-.85898E-02	0.
5	.1250	.3750	.10703E-01	.93182E-02	-.45473E-02	0.
6	.1250	.4718	.11671E-01	.10243E-01	-.53073E-03	0.
7	.2218	.2782	.14812E-01	.14846E-01	-.67909E-02	0.
8	.2218	.3750	.19151E-01	.16282E-01	-.37263E-02	0.
9	.2218	.4718	.17815E-01	.16015E-01	-.10205E-02	0.

ELEMENT NO. = 4

1	.3028	.3028	.17182E-01	.17182E-01	-.45805E-02	0.
2	.3028	.4472	.18488E-01	.18023E-01	-.33076E-02	0.
3	.4472	.3028	.21135E-01	.19267E-01	-.13109E-02	0.
4	.4472	.4472	.18023E-01	.18488E-01	-.33076E-02	0.
5	.3750	.3750	.20733E-01	.20733E-01	-.17880E-02	0.
6	.3750	.4718	.22807E-01	.22787E-01	-.23644E-03	0.
7	.4718	.2782	.19267E-01	.21135E-01	-.13109E-02	0.
8	.4718	.3750	.22787E-01	.22807E-01	-.23644E-03	0.
9	.4718	.4718	.23695E-01	.23695E-01	.17833E-03	0.

. . .

. . . .

. . . .

```
     INCREMENT NUMBER    30
     LOAD FACTOR =     .85600      CONVERGENCE TOLERANCE =    .10000     MAX. NO. OF ITERATIONS =   60
INITIAL OUTPUT PARAMETER =     0     FINAL OUTPUT PARAMETER =    3
IN CONVER            ITERATION NUMBER  1
DISPLACEMENT CHANGE NORM
  .282E+00       .280E+00       .280E+00
    TOTAL
   -.281E+00
RESIDUAL NORM
  .182E-10       .896E-06       .896E-06
    TOTAL
   -.605E-07
CONVERGENCE CODE   1
IN CONVER            ITERATION NUMBER  2
DISPLACEMENT CHANGE NORM
  .293E-06       .294E-06       .294E-06
    TOTAL
   -.294E-06
RESIDUAL NORM
  .183E-11       .266E-11       .245E-11
    TOTAL
   -.183E-11
CONVERGENCE CODE   0
     DISPLACEMENTS
      NODE       DISP.          XZ-ROT.        YZ-ROT.
         1  0.             0.             0.
         2  0.             0.              .455052E+04
         3  0.             0.              .879322E+04
         4  0.             0.              .106738E+05
         5  0.             0.              .118879E+05
         6  0.              .455052E+04   0.
         7  0.              .410180E+03    .410180E+03
         8   .101997E+04    .289215E+04    .742347E+04
         9  0.              .984409E+02    .623582E+03
        10   .139568E+04   0.              .102627E+05
        11  0.              .879322E+04   0.
        12   .101997E+04    .742347E+04    .289215E+04
        13   .183493E+04    .557560E+04    .557560E+04
        14   .235881E+04    .275674E+04    .688004E+04
        15   .252626E+04   0.              .772635E+04
```

16	0.	.106738E+05	0.
17	0.	.623582E+03	.984409E+02
18	.235881E+04	.688004E+04	.275674E+04
19	0.	.230744E+03	.230744E+03
20	.325803E+04	0.	.389260E+04
21	0.	.118879E+05	0.
22	.139568E+04	.102627E+05	0.
23	.252626E+04	.772635E+04	0.
24	.325803E+04	.389260E+04	0.
25	.349631E+04	0.	0.

REACTIONS

NODE	FORCE	XZ-MOMENT	YZ-MOMENT
1	.254174E-01	-.405413E-03	-.405413E-03
2	-.704030E-01	-.474595E-02	0.
3	.489298E-01	-.861086E-03	0.
4	-.130462E+00	-.178824E-02	0.
5	.322264E-01	-.228435E-02	0.
6	-.704030E-01	0.	-.474595E-02
10	0.	-.368943E-02	0.
11	.489298E-01	0.	-.861086E-03
15	0.	-.181699E-02	0.
16	-.130462E+00	0.	-.178824E-02
20	0.	-.720662E-02	0.
21	.322264E-01	0.	-.228435E-02
22	0.	0.	-.368943E-02
23	0.	0.	-.181699E-02
24	0.	0.	-.720662E-02
25	0.	-.132398E-02	-.132398E-02

STRESSES

G.P.	X-COORD.	Y-COORD.	X-MOMENT	Y-MOMENT	XY-MOMENT	EFF.PL.STRAIN
ELEMENT NO. = 1						
1	.0528	.0528	-.99908E-03	-.99908E-03	-.23087E-01	.57698E+04
2	.0528	.1972	.51873E-03	.14760E-02	-.23082E-01	.26193E+04
3	.1972	.0528	.80061E-02	.59482E-02	-.20218E-01	0.
4	.1972	.1972	.14760E-02	.51873E-03	-.23082E-01	.26193E+04
5	.1250	.1250	.86235E-02	.86235E-02	-.20786E-01	0.
6	.1250	.2218	.15459E-01	.16648E-01	-.16501E-01	0.
7	.2218	.0282	.59482E-02	.80061E-02	-.20218E-01	0.
8	.2218	.1250	.16648E-01	.15459E-01	-.16501E-01	0.
9	.2218	.2218	.21010E-01	.21010E-01	-.14518E-01	0.

```
  ELEMENT NO. =    2
 1     .3028     .0528  .43677E-02  .77768E-02 -.14262E-01 0.
 2     .3028     .1972  .14580E-01  .16625E-01 -.14808E-01 0.
 3     .4472     .0528  .25755E-01  .25762E-01 -.11744E-01 0.
 4     .4472     .1972  .36231E-02  .39462E-02 -.88176E-02 0.
 5     .3750     .1250  .16121E-01  .18645E-01 -.78343E-02 0.
 6     .3750     .2218  .28118E-01  .33194E-01 -.64873E-02 0.
 7     .4718     .0282  .55108E-02  .88899E-02 -.29126E-02 0.
 8     .4718     .1250  .17512E-01  .20166E-01 -.91283E-03 0.
 9     .4718     .2218  .27549E-01  .30853E-01 -.17952E-02 0.
  ELEMENT NO. =    3
 1     .0528     .3028  .77768E-02  .43677E-02 -.14262E-01 0.
 2     .0528     .4472  .39462E-02  .36231E-02 -.88176E-02 0.
 3     .1972     .3028  .88899E-02  .55108E-02 -.29126E-02 0.
 4     .1972     .4472  .16625E-01  .14580E-01 -.14808E-01 0.
 5     .1250     .3750  .18645E-01  .16121E-01 -.78343E-02 0.
 6     .1250     .4718  .20166E-01  .17512E-01 -.91283E-03 0.
 7     .2218     .2782  .25762E-01  .25755E-01 -.11744E-01 0.
 8     .2218     .3750  .33194E-01  .28118E-01 -.64873E-02 0.
 9     .2218     .4718  .30853E-01  .27549E-01 -.17952E-02 0.
  ELEMENT NO. =    4
 1     .3028     .3028  .29634E-01  .29634E-01 -.79751E-02 0.
 2     .3028     .4472  .31762E-01  .31040E-01 -.57935E-02 0.
 3     .4472     .3028  .36223E-01  .33145E-01 -.23092E-02 0.
 4     .4472     .4472  .31040E-01  .31762E-01 -.57935E-02 0.
 5     .3750     .3750  .35776E-01  .35776E-01 -.31804E-02 0.
 6     .3750     .4718  .39413E-01  .39460E-01 -.46572E-03 0.
 7     .4718     .2782  .33145E-01  .36223E-01 -.23092E-02 0.
 8     .4718     .3750  .39460E-01  .39413E-01 -.46572E-03 0.
 9     .4718     .4718  .39997E-01  .39997E-01  .26371E-03  .19186E+04
 1      .109200E+02  .300000E+00  .100000E-01  .100000E+01  0.          .400000E-01  0.          0.
CONVERGENCE PARAMETERS
IFDIS = 1     NCDIS =1110
IFRES = 1     NCRES =1110
 UNIFORMLY DISTRIBUTED LOADING INTENSITY -0.01LB/SQ INCH
 0
    TOTAL NODAL FORCES FOR EACH ELEMENT
 1       -.5208E-02  0.          0.          .2083E-01  0.          0.          -.5208E-02  0.
         0.          .2083E-01  0.          0.          -.5208E-02  0.          0.          .2083E-01
         0.          0.          -.5208E-02  0.          0.          .2083E-01  0.          0.
```

```
          .2778E-01  0.         0.
  2      -.5208E-02  0.         0.          .2083E-01  0.         0.        -.5208E-02  0.
          0.          .2083E-01  0.         0.        -.5208E-02  0.         0.          .2083E-01
          0.         0.        -.5208E-02  0.         0.          .2083E-01  0.         0.
          .2778E-01  0.         0.
  3      -.5208E-02  0.         0.          .2083E-01  0.         0.        -.5208E-02  0.
          0.          .2083E-01  0.         0.        -.5208E-02  0.         0.          .2083E-01
          0.         0.        -.5208E-02  0.         0.          .2083E-01  0.         0.
          .2778E-01  0.         0.
  4      -.5208E-02  0.         0.          .2083E-01  0.         0.        -.5208E-02  0.
          0.          .2083E-01  0.         0.        -.5208E-02  0.         0.          .2083E-01
          0.         0.        -.5208E-02  0.         0.          .2083E-01  0.         0.
          .2778E-01  0.         0.
     INCREMENT NUMBER     1
     LOAD FACTOR =    .50000     CONVERGENCE TOLERANCE =     .10000     MAX. NO. OF ITERATIONS =   60
INITIAL OUTPUT PARAMETER =    0     FINAL OUTPUT PARAMETER =    3
IN CONVER            ITERATION NUMBER  1
DISPLACEMENT CHANGE NORM
  .100E+03       .100E+03       .100E+03
    TOTAL
   -.100E+03
RESIDUAL NORM
  .845E-08       .662E-08       .628E-08
    TOTAL
   -.845E-08
CONVERGENCE CODE   1
IN CONVER            ITERATION NUMBER  2
DISPLACEMENT CHANGE NORM
  .918E-08       .908E-08       .897E-08
    TOTAL
   -.903E-08
RESIDUAL NORM
  .265E-11       .200E-11       .295E-11
    TOTAL
   -.265E-11
CONVERGENCE CODE   0
     DISPLACEMENTS
      NODE      DISP.         XZ-ROT.       YZ-ROT.
        1  0.            0.            0.
        2  0.            0.             .261614E+04
```

3	0.	0.	.505686E+04
4	0.	0.	.615962E+04
5	0.	0.	.687815E+04
6	0.	.261614E+04	0.
7	0.	.230157E+03	.230157E+03
8	.587914E+03	.167781E+04	.428957E+04
9	0.	.639453E+02	.362274E+03
10	.807234E+03	0.	.593553E+04
11	0.	.505686E+04	0.
12	.587914E+03	.428957E+04	.167781E+04
13	.105976E+04	.323511E+04	.323511E+04
14	.136395E+04	.160213E+04	.398710E+04
15	.146134E+04	0.	.447417E+04
16	0.	.615962E+04	0.
17	0.	.362274E+03	.639453E+02
18	.136395E+04	.398710E+04	.160213E+04
19	0.	.132888E+03	.132888E+03
20	.188400E+04	0.	.224070E+04
21	0.	.687815E+04	0.
22	.807234E+03	.593553E+04	0.
23	.146134E+04	.447417E+04	0.
24	.188400E+04	.224070E+04	0.
25	.202089E+04	0.	0.

REACTIONS

NODE	FORCE	XZ-MOMENT	YZ-MOMENT
1	.124667E-01	-.357597E-03	-.357597E-03
2	-.399935E-01	-.292695E-02	0.
3	.280665E-01	-.486232E-03	0.
4	-.754754E-01	-.103874E-02	0.
5	.186691E-01	-.132162E-02	0.
6	-.399935E-01	0.	-.292695E-02
10	0.	-.215061E-02	0.
11	.280665E-01	0.	-.486232E-03
15	0.	-.105925E-02	0.
16	-.754754E-01	0.	-.103874E-02
20	0.	-.417385E-02	0.
21	.186691E-01	0.	-.132162E-02
22	0.	0.	-.215061E-02
23	0.	0.	-.105925E-02
24	0.	0.	-.417385E-02
25	0.	-.783842E-03	-.783842E-03

```
        STRESSES
G.P.  X-COORD.   Y-COORD.    X-MOMENT     Y-MOMENT     XY-MOMENT    EFF.PL.STRAIN
        ELEMENT NO. =     1
   1       .0528      .0528 -.61926E-03 -.61926E-03 -.15278E-01 0.
   2       .0528      .1972  .27063E-03  .86852E-03 -.14197E-01 0.
   3       .1972      .0528  .44517E-02  .33436E-02 -.11677E-01 0.
   4       .1972      .1972  .86852E-03  .27063E-03 -.14197E-01 0.
   5       .1250      .1250  .48794E-02  .48794E-02 -.12011E-01 0.
   6       .1250      .2218  .87584E-02  .94485E-02 -.95831E-02 0.
   7       .2218      .0282  .33436E-02  .44517E-02 -.11677E-01 0.
   8       .2218      .1250  .94485E-02  .87584E-02 -.95831E-02 0.
   9       .2218      .2218  .11999E-01  .11999E-01 -.84458E-02 0.
        ELEMENT NO. =     2
   1       .3028      .0528  .24316E-02  .42854E-02 -.83085E-02 0.
   2       .3028      .1972  .83078E-02  .94492E-02 -.85898E-02 0.
          .          .          .          .
          .          .          .          .
          .          .          .          .
          .          .          .          .
```

etc.

A.4.8 Solution of dynamic transient elasto-plastic problem by explicit time integration. Example of Section 10.7.2, Fig. 10.3

Input data

```
53   10    2    1
SPHERICAL CAP EXAMPLE ,DYNPAK , SECTION 10.7.2 ,FIG. 10.3
 6    3    8   11    2    2    4    2    0    0    2    3    1
 1    1    1    4    6    7    8    5    3    2
 2    1    6    9   11   12   13   10    8    7
 3    1   11   14   16   17   18   15   13   12
 4    1   16   19   21   22   23   20   18   17
 5    1   21   24   26   27   28   25   23   22
 6    1   26   29   31   32   33   30   28   27
 7    1   31   34   36   37   38   35   33   32
 8    1   36   39   41   42   43   40   38   37
 9    1   41   44   46   47   48   45   43   42
10    1   46   49   51   52   53   50   48   47
 1      22.27    0.0000          32     22.475    16.0020
 4      22.27    1.3335          37     22.475    18.6690
 6      22.27    2.6670          42     22.475    21.3360
 9      22.27    4.0005          47     22.475    24.0030
```

```
11      22.27    5.3340
14      22.27    6.6675
16      22.27    8.0010
19      22.27    9.3345
21      22.27   10.6680
24      22.27   12.0015
26      22.27   13.3350
29      22.27   14.6685
31      22.27   16.0020
34      22.27   17.3355
36      22.27   18.6690
39      22.27   20.0025
41      22.27   21.3360
44      22.27   22.6695
46      22.27   24.0030
49      22.27   25.3365
51      22.27   26.6700
 2      22.475  00.0000
 7      22.475   2.6670
12      22.475   5.3340
17      22.475   8.0010
22      22.475  10.6680
27      22.475  13.3350
52      22.475  26.6700
 3      22.68   00.0000
 5      22.68    1.3335
 8      22.68    2.6670
10      22.68    4.0005
13      22.68    5.3340
15      22.68    6.6675
18      22.68    8.0010
20      22.68    9.3345
23      22.68   10.6680
25      22.68   12.0015
28      22.68   13.3350
30      22.68   14.6685
33      22.68   16.0020
35      22.68   17.3355
38      22.68   18.6690
40      22.68   20.0025
43      22.68   21.3360
45      22.68   22.6695
48      22.68   24.0030
50      22.68   25.3365
53      22.68   26.6700
    1   10
    2   10
    3   10
   51   11
   52   11
   53   11
    1
10500000.00.3     0.0        0.000245  0.0        24000.0   214285.71 0.0
10000.0   1.0     1.0
  500   10  250    1    1    1    1    0    0    0    1
0.0000004 0.001   0.0        0.0        0.0        0.0        0.0        0.0
0.0       0.0     0.0
    1
    1
    2    2    2    2    2    2    2    2    2    2
   53       0.0
   53       0.0
```

```
  DISTRIBUTED STEP PRESSURE P=600LB/IN SQ.
   0     0     1     0
  10
   1     8     5     3
600.0     600.0     600.0     0.0     0.0     0.0
   2    13    10     8
600.0     600.0     600.0     0.0     0.0     0.0
   3    18    15    13
600.0     600.0     600.0     0.0     0.0     0.0
   4    23    20    18
600.0     600.0     600.0     0.0     0.0     0.0
   5    28    25    23
600.0     600.0     600.0     0.0     0.0     0.0
   6    33    30    28
600.0     600.0     600.0     0.0     0.0     0.0
   7    38    35    33
600.0     600.0     600.0     0.0     0.0     0.0
   8    43    40    38
600.0     600.0     600.0     0.0     0.0     0.0
   9    48    45    43
600.0     600.0     600.0     0.0     0.0     0.0
  10    53    50    48
600.0     600.0     600.0     0.0     0.0     0.0
```

Line printer output

```
SPHERICAL CAP EXAMPLE ,DYNPAK ,SECTION 10.7.2 ,FIG. 10.3
 CONTROL PARAMETERS
  NPOIN =        53     NELEM =        10     NVFIX =         6
  NTYPE =         3     NNODE =         8     NDOFN =         2
  NMATS =         1     NPROP =        11     NGAUS =         2
  NDIME =         2     NSTRE =         4     NCRIT =         2
  NPREV =         0     NCONM =         0     NLAPS =         2
  NGAUM =         3     NRADS =         1
```

ELEMENT	PROPERTY	NODE NUMBERS							
1	1	1	4	6	7	8	5	3	2
2	1	6	9	11	12	13	10	8	7
3	1	11	14	16	17	18	15	13	12
4	1	16	19	21	22	23	20	18	17
5	1	21	24	26	27	28	25	23	22
6	1	26	29	31	32	33	30	28	27
7	1	31	34	36	37	38	35	33	32
8	1	36	39	41	42	43	40	38	37
9	1	41	44	46	47	48	45	43	42
10	1	46	49	51	52	53	50	48	47

1	22.270	0.000
4	22.270	1.334
6	22.270	2.667
9	22.270	4.001
11	22.270	5.334
14	22.270	6.668
16	22.270	8.001
19	22.270	9.335
21	22.270	10.668
24	22.270	12.002
26	22.270	13.335
29	22.270	14.669
31	22.270	16.002
34	22.270	17.336
36	22.270	18.669
39	22.270	20.003
41	22.270	21.336
44	22.270	22.670
46	22.270	24.003
49	22.270	25.337
51	22.270	26.670
2	22.475	0.000
7	22.475	2.667
12	22.475	5.334
17	22.475	8.001
22	22.475	10.668
27	22.475	13.335
32	22.475	16.002
37	22.475	18.669

NODE	X	Y
1	0.000	22.270
2	0.000	22.475
3	0.000	22.680
4	.518	22.264
5	.528	22.674
6	1.036	22.246
7	1.046	22.451
8	1.055	22.655
9	1.554	22.216
10	1.582	22.625
11	2.070	22.174
12	2.089	22.378
13	2.108	22.582
14	2.586	22.119
15	2.633	22.527
16	3.100	22.053
17	3.128	22.256
18	3.157	22.459
19	3.612	21.975
20	3.679	22.380
21	4.123	21.885
22	4.161	22.087
23	4.198	22.288
24	4.631	21.783
25	4.716	22.184
26	5.136	21.670
27	5.184	21.869
28	5.231	22.069

42	22.475	21.336
47	22.475	24.003
52	22.475	26.670
3	22.680	0.000
5	22.680	1.334
8	22.680	2.667
10	22.680	4.001
13	22.680	5.334
15	22.680	6.668
18	22.680	8.001
20	22.680	9.335
23	22.680	10.668
25	22.680	12.002
28	22.680	13.335
30	22.680	14.669
33	22.680	16.002
35	22.680	17.336
38	22.680	18.669
40	22.680	20.003
43	22.680	21.336
45	22.680	22.670
48	22.680	24.003
50	22.680	25.337
53	22.680	26.670

29	5.639	21.544
30	5.743	21.941
31	6.139	21.407
32	6.196	21.604
33	6.252	21.801
34	6.636	21.258
35	6.758	21.650
36	7.129	21.098
37	7.194	21.292
38	7.260	21.487
39	7.618	20.927
40	7.758	21.312
41	8.103	20.744
42	8.177	20.935
43	8.252	21.126
44	8.583	20.549
45	8.741	20.928
46	9.059	20.344
47	9.142	20.531
48	9.226	20.719
49	9.530	20.128
50	9.706	20.498
51	9.996	19.901
52	10.088	20.084
53	10.180	20.267

NODE	CODE
1	10
2	10
3	10
4	00
5	00
6	00
7	00
8	00
9	00
10	00
11	00
12	00
13	00
14	00
15	00
16	00
17	00
18	00
19	00
20	00
21	00
22	00
23	00
24	00
25	00
26	00
27	00
28	00
29	00
30	00
31	00
32	00
33	00
34	00
35	00
36	00
37	00
38	00
39	00
40	00
41	00
42	00
43	00
44	00
45	00
46	00
47	00
48	00
49	00
50	00
51	11
52	11
53	11

```
MATERIAL PROPERTIES
MATERIAL NO     1
YOUNG MODULUS    .1050E+08
POISSON RATIO    .3000
THICKNESS       0.
MASS DENSITY     .2450E-03
ALPHA TEMPR     0.
REFERENCE FO     .2400E+05
HARDENING PAR    .2143E+06
FRICT ANGLE     0.
FLUIDITY PAR     .1000E+05
EXP DELTA        1.000
NFLOW CODE       1.000
TIME STEPPING PARAMETERS
NSTEP=  500           NOUTD=   10           NOUTP=  250
NREQD=    1           NREQS=    1           NACCE=    1
IFUNC=    1           IFIXD=    0           MITER=    0
KSTEP=    0           IPRED=    1
DTIME=  .4000E-06     DTEND=  .1000E-02     DTREC=  0.
AALFA=  0.            BEETA=  0.            DELTA=  0.
GAAMA=  0.            AZERO=  0.            BZERO=  0.
OMEGA=  0.            TOLER=  0.
 SELECTIVE OUTPUT REQUESTED FOR FOLLOWING
 NODES     1
 G.P.      1
  TYPE OF ELEMENT, IMPLICIT=1,EXPLICIT=2
2    2    2    2    2    2    2    2    2    2
 NODE   INITIAL X-DISP.   INITIAL Y-DISP.
   53       0.                 0.
 NODE   INITIAL X-VELO.   INITIAL Y-VELO.
   53       0.                 0.
LOAD CASE TITLE -   DISTRIBUTED STEP PRESSURE P=600LB/IN
LOAD INPUT PARAMETERS
POINT LOADS      0
GRAVITY          0
EDGE LOAD        1
TEMPERATURE      0
 NO. OF LOADED EDGES =   10
 LIST OF LOADED EDGES AND APPLIED LOADS
    1         8    5    3
```

```
600.000   600.000   600.000     0.000     0.000     0.000
      2        13   10    8
600.000   600.000   600.000     0.000     0.000     0.000
      3        18   15   13
600.000   600.000   600.000     0.000     0.000     0.000
      4        23   20   18
600.000   600.000   600.000     0.000     0.000     0.000
      5        28   25   23
600.000   600.000   600.000     0.000     0.000     0.000
      6        33   30   28
600.000   600.000   600.000     0.000     0.000     0.000
      7        38   35   33
600.000   600.000   600.000     0.000     0.000     0.000
      8        43   40   38
600.000   600.000   600.000     0.000     0.000     0.000
      9        48   45   43
600.000   600.000   600.000     0.000     0.000     0.000
     10        53   50   48
600.000   600.000   600.000     0.000     0.000     0.000
```

NODAL LUMPED MASSES

1	.10000E+31	2	.90632E-05	3	.10000E+31	4	.36354E-04	5	.10000E+31	6	.91129E-05
7	.72039E-04	8	.72039E-04	9	.73365E-04	10	.73365E-04	11	.54175E-04	12	.54175E-04
13	.29072E-03	14	.29072E-03	15	.54838E-04	16	.54838E-04	17	.21596E-03	18	.21596E-03
19	.21994E-03	20	.21994E-03	21	.10823E-03	22	.10823E-03	23	.58081E-03	24	.58081E-03
25	.10956E-03	26	.10956E-03	27	.35941E-03	28	.35941E-03	29	.36603E-03	30	.36603E-03
31	.16206E-03	32	.16206E-03	33	.86965E-03	34	.86965E-03	35	.16404E-03	36	.16404E-03
37	.50209E-03	38	.50209E-03	39	.51133E-03	40	.51133E-03	41	.21553E-03	42	.21553E-03
43	.11566E-02	44	.11566E-02	45	.21816E-03	46	.21816E-03	47	.64368E-03	48	.64368E-03
49	.65553E-03	50	.65553E-03	51	.26853E-03	52	.26853E-03	53	.14410E-02	54	.14410E-02
55	.27182E-03	56	.27182E-03	57	.78387E-03	58	.78387E-03	59	.79830E-03	60	.79830E-03
61	.32096E-03	62	.32096E-03	63	.17224E-02	64	.17224E-02	65	.32488E-03	66	.32488E-03
67	.92236E-03	68	.92236E-03	69	.93934E-03	70	.93934E-03	71	.37268E-03	72	.37268E-03
73	.20000E-02	74	.20000E-02	75	.37724E-03	76	.37724E-03	77	.10589E-02	78	.10589E-02
79	.10784E-02	80	.10784E-02	81	.42360E-03	82	.42360E-03	83	.22732E-02	84	.22732E-02
85	.42879E-03	86	.42879E-03	87	.11931E-02	88	.11931E-02	89	.12150E-02	90	.12150E-02
91	.47361E-03	92	.47361E-03	93	.25415E-02	94	.25415E-02	95	.47940E-03	96	.47940E-03
97	.13247E-02	98	.13247E-02	99	.13491E-02	100	.13491E-02	101	.10000E+31	102	.10000E+31
103	.10000E+31	104	.10000E+31	105	.10000E+31	106	.10000E+31				

```
 DISPLACEMENTS AT TIME STEP        250     TIME     .10000000000E-03
NNODE   X-DISP      Y-DISP    NNODE   X-DISP      Y-DISP    NNODE   X-DISP      Y-DISP
```

1	-.80217E-36	-.24592E-01	2	.16169E-37	-.24444E-01	3	.80603E-36	-.24278E-01
4	-.48654E-03	-.24378E-01	5	-.47049E-03	-.24082E-01	6	-.10271E-02	-.24452E-01
7	-.95811E-03	-.24277E-01	8	-.89235E-03	-.24136E-01	9	-.16057E-02	-.24657E-01
10	-.12997E-02	-.24370E-01	11	-.21111E-02	-.25054E-01	12	-.19625E-02	-.24907E-01
13	-.18173E-02	-.24753E-01	14	-.24674E-02	-.25202E-01	15	-.24494E-02	-.24886E-01
16	-.27549E-02	-.24939E-01	17	-.29046E-02	-.24767E-01	18	-.30484E-02	-.24575E-01
19	-.31689E-02	-.24317E-01	20	-.34286E-02	-.23956E-01	21	-.38241E-02	-.23993E-01
22	-.37338E-02	-.23818E-01	23	-.36581E-02	-.23704E-01	24	-.46693E-02	-.24321E-01
25	-.39541E-02	-.24145E-01	26	-.55750E-02	-.25293E-01	27	-.50270E-02	-.25239E-01
28	-.44904E-02	-.25223E-01	29	-.63674E-02	-.26563E-01	30	-.53025E-02	-.26494E-01
31	-.68354E-02	-.27549E-01	32	-.65796E-02	-.27490E-01	33	-.63050E-02	-.27373E-01
34	-.68148E-02	-.27616E-01	35	-.72371E-02	-.27120E-01	36	-.61813E-02	-.26275E-01
37	-.70445E-02	-.25897E-01	38	-.78704E-02	-.25402E-01	39	-.49243E-02	-.23242E-01
40	-.78642E-02	-.21832E-01	41	-.31809E-02	-.18638E-01	42	-.51813E-02	-.17807E-01
43	-.71315E-02	-.16874E-01	44	-.13257E-02	-.13014E-01	45	-.55830E-02	-.10958E-01
46	.15858E-03	-.72532E-02	47	-.16481E-02	-.63066E-02	48	-.35144E-02	-.55039E-02
49	.71545E-03	-.26453E-02	50	-.14109E-02	-.13718E-02	51	.17196E-33	-.61722E-33
52	.47545E-33	.53008E-33	53	-.45643E-33	-.26357E-33	54	0.	0.

STRESSES

G.P.	RR-STRESS	ZZ-STRESS	RZ-STRESS	TT-STRESS	MAX P.S.	MIN P.S.	ANGLE	P.S.
ELEMENT NO. =	1							
1	-.142403E+05	-.105408E+04	.188251E+04	-.141343E+05	-.790586E+03	-.145038E+05	-7.968	0.
2	-.137476E+05	-.359556E+03	.175780E+04	-.138344E+05	-.132611E+03	-.139746E+05	-7.357	0.
3	-.147863E+05	-.306845E+03	.360377E+03	-.144528E+05	-.297881E+03	-.147952E+05	-1.425	0.
4	-.130724E+05	-.794640E+03	.369614E+03	-.134549E+05	-.783523E+03	-.130835E+05	-1.723	0.
ELEMENT NO. =	2							
1	-.152776E+05	-.476674E+03	.112943E+04	-.149803E+05	-.390986E+03	-.153633E+05	-4.339	0.
2	-.123991E+05	-.419294E+03	.854064E+03	-.129006E+05	-.358713E+03	-.124596E+05	-4.057	0.
3	-.139979E+05	-.424122E+03	.544093E+03	-.146986E+05	-.402347E+03	-.140197E+05	-2.292	0.
4	-.137154E+05	-.101627E+03	.580061E+03	-.133725E+05	-.769561E+02	-.137401E+05	-2.435	0.
ELEMENT NO. =	3							
1	-.120725E+05	.859252E+02	.104211E+04	-.137897E+05	.174599E+03	-.121612E+05	-4.864	0.
2	-.155576E+05	-.672336E+03	.121601E+04	-.144822E+05	-.573651E+03	-.156563E+05	-4.640	0.
3	-.115157E+05	-.754333E+02	.151423E+04	-.131835E+05	.121596E+03	-.117127E+05	-7.414	0.
4	-.158138E+05	-.995668E+03	.202746E+04	-.149737E+05	-.723272E+03	-.160862E+05	-7.652	0.
ELEMENT NO. =	4							
1	-.133486E+05	-.746264E+03	.285467E+04	-.135909E+05	-.129786E+03	-.139651E+05	-12.186	0.
2	-.133564E+05	-.736688E+03	.284774E+04	-.140809E+05	-.123832E+03	-.139692E+05	-12.145	0.
3	-.161873E+05	-.716272E+03	.322767E+04	-.145098E+05	-.698999E+02	-.168337E+05	-11.324	0.
4	-.105957E+05	-.963920E+03	.210077E+04	-.129019E+05	-.525664E+03	-.110339E+05	-11.784	0.

ELEMENT NO. =	5							
1	-.168527E+05	-.712967E+03	.355630E+04	-.151044E+05	.358997E+02	-.176015E+05	-11.891	0.
2	-.965035E+04	-.718838E+03	.203275E+04	-.122643E+05	-.277959E+03	-.100912E+05	-12.237	0.
3	-.157929E+05	-.601834E+03	.309818E+04	-.154933E+05	.573147E+01	-.164005E+05	-11.095	0.
4	-.109494E+05	-.556352E+03	.189021E+04	-.127634E+05	-.223251E+03	-.112825E+05	-9.994	0.
ELEMENT NO. =	6							
1	-.136993E+05	-.430264E+03	.276735E+04	-.153917E+05	.123755E+03	-.142533E+05	-11.321	0.
2	-.130603E+05	-.829663E+03	.258448E+04	-.139319E+05	-.305957E+03	-.135840E+05	-11.455	0.
3	-.100891E+05	-.428639E+03	.185225E+04	-.146315E+05	-.856752E+02	-.104321E+05	-10.490	0.
4	-.167170E+05	-.106309E+04	.358040E+04	-.158653E+05	-.283044E+03	-.174970E+05	-12.291	0.
ELEMENT NO. =	7							
1	-.748198E+04	-.154126E+03	.155617E+04	-.136080E+05	.162653E+03	-.779876E+04	-11.506	0.
2	-.188538E+05	-.171938E+04	.502582E+04	-.172146E+05	-.354022E+03	-.202192E+05	-15.199	0.
3	-.498918E+04	-.283243E+03	.123802E+04	-.118093E+05	.225769E+02	-.529500E+04	-13.876	0.
4	-.211022E+05	-.239729E+04	.599980E+04	-.181324E+05	-.638225E+03	-.228613E+05	-16.340	0.
ELEMENT NO. =	8							
1	-.421590E+04	-.332032E+03	.167829E+04	-.101971E+05	.292698E+03	-.484063E+04	-20.417	0.
2	-.203671E+05	-.292740E+04	.739486E+04	-.175571E+05	-.213976E+03	-.230805E+05	-20.150	0.
3	-.579043E+04	-.191795E+04	.260848E+04	-.859256E+04	-.605623E+03	-.710276E+04	-26.707	0.
4	-.179717E+05	-.280583E+04	.716447E+04	-.151334E+05	.434134E+02	-.208210E+05	-21.687	0.
ELEMENT NO. =	9							
1	-.808480E+04	-.179792E+04	.441523E+04	-.705104E+04	.478552E+03	-.103613E+05	-27.275	0.
2	-.138434E+05	-.337445E+04	.706677E+04	-.122206E+05	.185344E+03	-.174031E+05	-26.736	0.
3	-.126711E+05	-.279604E+04	.796913E+04	-.613091E+04	.164120E+04	-.171083E+05	-29.109	0.
4	-.825151E+04	-.490425E+04	.337893E+04	-.813863E+04	-.280717E+04	-.103486E+05	-31.825	0.
ELEMENT NO. =	10							
1	-.175308E+05	-.460688E+04	.983514E+04	-.674090E+04	.699203E+03	-.228369E+05	-28.347	0.
2	-.149784E+04	-.237914E+04	.298633E+04	-.325566E+04	.108018E+04	-.495715E+04	40.803	0.
3	-.253662E+05	-.151577E+05	.109573E+05	-.118829E+05	-.817417E+04	-.323498E+05	-32.511	0.
4	.721668E+04	.453440E+04	.661719E+03	.320558E+04	.737105E+04	.438003E+04	13.131	0.

. . . .

. . . .

. . . .

. . . .

etc.

A.4.9 Solution of dynamic transient elasto-plastic problem by implicit/explicit approach. Example of Section 11.6.1, Fig. 11.4

Input data

```
53   10    2    1
SPHERICAL CAP EXAMPLE ,MIXDYN ,SECTION 11.6.1 ,FIG. 11.4
 6    3    8   11    2    2    4    2    0    0    2    3    1
 1    1    1    4    6    7    8    5    3    2
 2    1    6    9   11   12   13   10    8    7
 3    1   11   14   16   17   18   15   13   12
 4    1   16   19   21   22   23   20   18   17
 5    1   21   24   26   27   28   25   23   22
 6    1   26   29   31   32   33   30   28   27
 7    1   31   34   36   37   38   35   33   32
 8    1   36   39   41   42   43   40   38   37
 9    1   41   44   46   47   48   45   43   42
10    1   46   49   51   52   53   50   48   47

 1     22.27    0.0000       26     22.27   13.3350
 4     22.27    1.3335       29     22.27   14.6685
 6     22.27    2.6670       31     22.27   16.0020
 9     22.27    4.0005       34     22.27   17.3355
11     22.27    5.3340       36     22.27   18.6690
14     22.27    6.6675       39     22.27   20.0025
16     22.27    8.0010       41     22.27   21.3360
19     22.27    9.3345       44     22.27   22.6695
21     22.27   10.6680       46     22.27   24.0030
24     22.27   12.0015       49     22.27   25.3365
                             51     22.27   26.6700
```

```
  2   22.475   00.0000          15   22.68    6.6675
  7   22.475    2.6670          18   22.68    8.0010
 12   22.475    5.3340          20   22.68    9.3345
 17   22.475    8.0010          23   22.68   10.6680
 22   22.475   10.6680          25   22.68   12.0015
 27   22.475   13.3350          28   22.68   13.3350
 32   22.475   16.0020          30   22.68   14.6685
 37   22.475   18.6690          33   22.68   16.0020
 42   22.475   21.3360          35   22.68   17.3355
 47   22.475   24.0030          38   22.68   18.6690
 52   22.475   26.6700          40   22.68   20.0025
  3   22.68    00.0000          43   22.68   21.3360
  5   22.68     1.3335          45   22.68   22.6695
  8   22.68     2.6670          48   22.68   24.0030
 10   22.68     4.0005          50   22.68   25.3365
 13   22.68     5.3340          53   22.68   26.6700
    1   10
    2   10
    3   10
   51   11
   52   11
   53   11
    1
10500000.00.3      0.0       0.000245  0.0       24000.0   214285.71 0.0
10000.0   1.0      1.0
  200    1   20    1    1    1    1    0    5  201    2
0.0000050 0.001    0.0       0.0       0.0       0.2500    0.50      0.0
0.0       0.0      0.00010000
    1
    1
    1    1    1    1    1    1    1    1    1    1
   53       0.0
   53       0.0
   DISTRIBUTED STEP PRESSURE P=600LB/IN SQ.
    0    0    1    0
   10
    1    8    5    3
600.0     600.0    600.0     0.0       0.0       0.0
    2   13   10    8
```

```
600.0    600.0    600.0    0.0    0.0    0.0
    3   18   15   13
600.0    600.0    600.0    0.0    0.0    0.0
    4   23   20   18
600.0    600.0    600.0    0.0    0.0    0.0
    5   28   25   23
600.0    600.0    600.0    0.0    0.0    0.0
    6   33   30   28
600.0    600.0    600.0    0.0    0.0    0.0
    7   38   35   33
600.0    600.0    600.0    0.0    0.0    0.0
    8   43   40   38
600.0    600.0    600.0    0.0    0.0    0.0
    9   48   45   43
600.0    600.0    600.0    0.0    0.0    0.0
   10   53   50   48
600.0    600.0    600.0    0.0    0.0    0.0
```

Line printer output

```
SPHERICAL CAP EXAMPLE , MIXDYN ,SECTION 11.6.1 ,FIG. 11.4
CONTROL PARAMETERS
 NPOIN =        53     NELEM =        10     NVFIX =          6
 NTYPE =         3     NNODE =         8     NDOFN =          2
 NMATS =         1     NPROP =        11     NGAUS =          2
 NDIME =         2     NSTRE =         4     NCRIT =          2
 NPREV =         0     NCONM =         0     NLAPS =          2
 NGAUM =         3     NRADS =         1
 ELEMENT   PROPERTY     NODE NUMBERS
     1         1         1    4    6    7    8    5    3    2
     2         1         6    9   11   12   13   10    8    7
     3         1        11   14   16   17   18   15   13   12
     4         1        16   19   21   22   23   20   18   17
     5         1        21   24   26   27   28   25   23   22
     6         1        26   29   31   32   33   30   28   27
     7         1        31   34   36   37   38   35   33   32
     8         1        36   39   41   42   43   40   38   37
     9         1        41   44   46   47   48   45   43   42
    10         1        46   49   51   52   53   50   48   47
```

1	22.270	0.000
4	22.270	1.334
6	22.270	2.667
9	22.270	4.001
11	22.270	5.334
14	22.270	6.668
16	22.270	8.001
19	22.270	9.335
21	22.270	10.668
24	22.270	12.002
26	22.270	13.335
29	22.270	14.669
31	22.270	16.002
34	22.270	17.336
36	22.270	18.669
39	22.270	20.003
41	22.270	21.336
44	22.270	22.670
46	22.270	24.003
49	22.270	25.337
51	22.270	26.670
2	22.475	0.000
7	22.475	2.667
12	22.475	5.334
17	22.475	8.001
22	22.475	10.668
27	22.475	13.335
32	22.475	16.002
37	22.475	18.669
42	22.475	21.336
47	22.475	24.003
52	22.475	26.670
3	22.680	0.000
5	22.680	1.334
8	22.680	2.667
10	22.680	4.001
13	22.680	5.334
15	22.680	6.668
18	22.680	8.001
20	22.680	9.335
23	22.680	10.668
25	22.680	12.002
28	22.680	13.335
30	22.680	14.669
33	22.680	16.002
35	22.680	17.336
38	22.680	18.669
40	22.680	20.003
43	22.680	21.336
45	22.680	22.670
48	22.680	24.003
50	22.680	25.337
53	22.680	26.670

NODE	X	Y
1	0.000	22.270
2	0.000	22.475
3	0.000	22.680
4	.518	22.264
5	.528	22.674
6	1.036	22.246
7	1.046	22.451
8	1.055	22.655
9	1.554	22.216
10	1.582	22.625
11	2.070	22.174
12	2.089	22.378
13	2.108	22.582
14	2.586	22.119
15	2.633	22.527
16	3.100	22.053
17	3.128	22.256
18	3.157	22.459
19	3.612	21.975
20	3.679	22.380
21	4.123	21.885
22	4.161	22.087
23	4.198	22.288
24	4.631	21.783

25	4.716	22.184
26	5.136	21.670
27	5.184	21.869
28	5.231	22.069
29	5.639	21.544
30	5.743	21.941
31	6.139	21.407
32	6.196	21.604
33	6.252	21.801
34	6.636	21.258
35	6.758	21.650
36	7.129	21.098
37	7.194	21.292
38	7.260	21.487
39	7.618	20.927
40	7.758	21.312
41	8.103	20.744
42	8.177	20.935
43	8.252	21.126
44	8.583	20.549
45	8.741	20.928
46	9.059	20.344
47	9.142	20.531
48	9.226	20.719
49	9.530	20.128
50	9.706	20.498
51	9.996	19.901
52	10.088	20.084
53	10.180	20.267

NŌDE	CODE
1	10
2	10
3	10
4	00
5	00
6	00
7	00
8	00
9	00
10	00
11	00
12	00
13	00
14	00
15	00
16	00
17	00
18	00
19	00
20	00
21	00
22	00
23	00
24	00
25	00
26	00
27	00
28	00
29	00
30	00
31	00
32	00
33	00
34	00
35	00
36	00
37	00
38	00
39	00
40	00
41	00
42	00
43	00
44	00
45	00
46	00
47	00
48	00
49	00
50	00
51	11
52	11
53	11

MATERIAL PROPERTIES
MATERIAL NO 1
YOUNG MODULUS .1050E+08
POISSON RATIO .3000
THICKNESS 0.

```
 MASS DENSITY    .2450E-03
 ALPHA TEMPR    0.
 REFERENCE FO    .2400E+05
 HARDENING PAR   .2143E+06
 FRICT ANGLE    0.
 FLUIDITY PAR    .1000E+05
 EXP DELTA       1.000
 NFLOW CODE      1.000
 TIME STEPPING PARAMETERS
 NSTEP=  200            NOUTD=     1          NOUTP=   20
 NREQD=    1            NREQS=     1          NACCE=    1
 IFUNC=    1            IFIXD=     0          MITER=    5
 KSTEP=  201            IPRED=     2
 DTIME=   .5000E-05     DTEND=    .1000E-02   DTREC=  0.
 AALFA=  0.             BEETA=  0.            DELTA=    .2500
 GAAMA=   .5000         AZERO=  0.            BZERO=  0.
 OMEGA=  0.             TOLER=    .1000E-03
  SELECTIVE OUTPUT REQUESTED FOR FOLLOWING
  NODES     1
  G.P.      1
   TYPE OF ELEMENT, IMPLICIT=1,EXPLICIT=2
1    1    1    1    1    1    1    1    1    1
  NODE   INITIAL X-DISP.   INITIAL Y-DISP.
    53       0.                 0.
  NODE   INITIAL X-VELO.   INITIAL Y-VELO.
    53       0.                 0.
 LOAD CASE TITLE -   DISTRIBUTED STEP PRESSURE P=600LB/IN
 LOAD INPUT PARAMETERS
 POINT LOADS      0
 GRAVITY          0
 EDGE LOAD        1
 TEMPERATURE      0
  NO. OF LOADED EDGES =    10
  LIST OF LOADED EDGES AND APPLIED LOADS
      1         8    5    3
600.000   600.000   600.000     0.000     0.000     0.000
      2        13   10    8
600.000   600.000   600.000     0.000     0.000     0.000
      3        18   15   13
600.000   600.000   600.000     0.000     0.000     0.000
      4        23   20   18
```

```
 600.000   600.000   600.000     0.000     0.000     0.000
       5        28    25   23
 600.000   600.000   600.000     0.000     0.000     0.000
       6        33    30   28
 600.000   600.000   600.000     0.000     0.000     0.000
       7        38    35   33
 600.000   600.000   600.000     0.000     0.000     0.000
       8        43    40   38
 600.000   600.000   600.000     0.000     0.000     0.000
       9        48    45   43
 600.000   600.000   600.000     0.000     0.000     0.000
      10        53    50   48
 600.000   600.000   600.000     0.000     0.000     0.000
   NEQNS=   97     NWMTL= 1045     NWKTL= 1045
       1    2    4    7   11   16   22   29   37   46   56   67   79   92   99  107  116  126  137  149
     162  176  191  207  214  222  231  241  252  264  277  291  306  322  329  337  346  356  367  379
     392  406  421  437  444  452  461  471  482  494  507  521  536  552  559  567  576  586  597  609
     622  636  651  667  674  682  691  701  712  724  737  751  766  782  789  797  806  816  827  839
     852  866  881  897  904  912  921  931  942  954  967  981  996 1012 1019 1027 1036 1046
       1    2    4    7   11   16   22   29   37   46   56   67   79   92   99  107  116  126  137  149
     162  176  191  207  214  222  231  241  252  264  277  291  306  322  329  337  346  356  367  379
     392  406  421  437  444  452  461  471  482  494  507  521  536  552  559  567  576  586  597  609
     622  636  651  667  674  682  691  701  712  724  737  751  766  782  789  797  806  816  827  839
     852  866  881  897  904  912  921  931  942  954  967  981  996 1012 1019 1027 1036 1046
INITIAL ACCELERATION
-.18236E+08  .91173E+07 -.54133E+08 -.35754E+06 -.18247E+08 -.13920E+07 -.54061E+08 -.93018E+06 -.18228E+08  .46432E+06
 .91132E+07 -.23578E+07 -.54047E+08 -.12424E+07 -.18205E+08 -.38329E+07 -.53943E+08 -.17443E+07 -.18168E+08  .87161E+06
 .90830E+07 -.49321E+07 -.53867E+08 -.21006E+07 -.18128E+08 -.63175E+07 -.53711E+08 -.25752E+07 -.18076E+08  .12872E+07
 .90371E+07 -.74684E+07 -.53581E+08 -.29436E+07 -.18003E+08 -.87955E+07 -.53354E+08 -.33945E+07 -.17917E+08  .16969E+07
 .89579E+07 -.99591E+07 -.53151E+08 -.37934E+07 -.17869E+08 -.11275E+08 -.52913E+08 -.42561E+07 -.17812E+08  .21275E+07
 .89049E+07 -.12464E+08 -.52692E+08 -.45797E+07 -.17590E+08 -.13681E+08 -.52254E+08 -.49619E+07 -.17373E+08  .24809E+07
 .86861E+07 -.14781E+08 -.51805E+08 -.55366E+07 -.17632E+08 -.16247E+08 -.51858E+08 -.62011E+07 -.17977E+08  .30987E+07
 .89849E+07 -.17535E+08 -.51741E+08 -.58186E+07 -.16318E+08 -.18180E+08 -.50157E+08 -.57372E+07 -.15178E+08  .28705E+07
 .75929E+07 -.18229E+08 -.47544E+08 -.82556E+07 -.19283E+08 -.22510E+08 -.53199E+08 -.12268E+08 -.26298E+08  .61105E+07
 .13102E+08 -.23831E+08 -.53089E+08 -.20703E+07 -.50752E+07 -.19969E+08 -.42545E+08
2 6 6 5 5 5 5 5 4 4 4 4 4 3 4 4 4 4 3 4
   DISPLACEMENTS AT TIME STEP        20     TIME     .10000000000E-03
NNODE   X-DISP      Y-DISP     NNODE   X-DISP       Y-DISP     NNODE   X-DISP       Y-DISP
    1 0.            -.24848E-01    2 0.            -.24695E-01    3 0.            -.24531E-01
    4 -.49085E-03 -.24866E-01      5 -.47375E-03 -.24549E-01      6 -.10248E-02 -.24900E-01
```

7	-.96448E-03	-.24721E-01	8	-.90720E-03	-.24585E-01	9	-.16044E-02	-.25115E-01
10	-.13198E-02	-.24815E-01	11	-.20699E-02	-.25440E-01	12	-.19725E-02	-.25296E-01
13	-.18734E-02	-.25137E-01	14	-.23709E-02	-.25414E-01	15	-.25289E-02	-.25071E-01
16	-.26710E-02	-.24905E-01	17	-.28704E-02	-.24721E-01	18	-.30683E-02	-.24530E-01
19	-.31289E-02	-.24240E-01	20	-.33998E-02	-.23877E-01	21	-.37999E-02	-.23915E-01
22	-.37049E-02	-.23738E-01	23	-.36218E-02	-.23626E-01	24	-.46539E-02	-.24268E-01
25	-.39278E-02	-.24100E-01	26	-.55678E-02	-.25249E-01	27	-.50092E-02	-.25194E-01
28	-.44634E-02	-.25182E-01	29	-.63749E-02	-.26559E-01	30	-.52828E-02	-.26501E-01
31	-.68507E-02	-.27577E-01	32	-.65811E-02	-.27518E-01	33	-.62954E-02	-.27401E-01
34	-.68161E-02	-.27664E-01	35	-.72519E-02	-.27166E-01	36	-.61665E-02	-.26284E-01
37	-.70370E-02	-.25898E-01	38	-.78691E-02	-.25398E-01	39	-.49158E-02	-.23247E-01
40	-.78646E-02	-.21836E-01	41	-.31688E-02	-.18623E-01	42	-.51687E-02	-.17788E-01
43	-.71196E-02	-.16855E-01	44	-.13242E-02	-.13015E-01	45	-.55782E-02	-.10963E-01
46	.15487E-03	-.72526E-02	47	-.16405E-02	-.62975E-02	48	-.35098E-02	-.54896E-02
49	.71525E-03	-.26446E-02	50	-.14080E-02	-.13752E-02	51	0.	0.
52	0.	0.	53	0.	0.	54	0.	0.

STRESSES

G.P.	RR-STRESS	ZZ-STRESS	RZ-STRESS	TT-STRESS	MAX P.S.	MIN P.S.	ANGLE	P.S.
ELEMENT NO. =		1						
1	-.140401E+05	-.297607E+03	.102996E+02	-.139518E+05	-.297599E+03	-.140401E+05	-.043	0.
2	-.137677E+05	-.163963E+03	-.597302E+01	-.138441E+05	-.163961E+03	-.137677E+05	.025	0.
3	-.144323E+05	.442655E+03	.738033E+03	-.141613E+05	.479184E+03	-.144688E+05	-2.834	0.
4	-.133330E+05	-.936998E+03	.632685E+03	-.136731E+05	-.904790E+03	-.133652E+05	-2.914	0.
ELEMENT NO. =		2						
1	-.152286E+05	-.445944E+02	.890663E+03	-.148846E+05	.747159E+01	-.152807E+05	-3.346	0.
2	-.125683E+05	-.504156E+03	.703159E+03	-.130704E+05	-.463311E+03	-.126091E+05	-3.324	0.
3	-.135150E+05	-.291803E+03	.500085E+03	-.144796E+05	-.272918E+03	-.135339E+05	-2.163	0.
4	-.143944E+05	-.155896E+03	.601210E+03	-.137405E+05	-.130555E+03	-.144197E+05	-2.414	0.
ELEMENT NO. =		3						
1	-.114077E+05	.321093E+03	.890750E+03	-.133350E+05	.388356E+03	-.114749E+05	-4.318	0.
2	-.163302E+05	-.869776E+03	.130850E+04	-.149798E+05	-.759812E+03	-.164402E+05	-4.804	0.
3	-.119282E+05	-.302176E+02	.202031E+04	-.130545E+05	.303478E+03	-.122619E+05	-9.379	0.
4	-.153434E+05	-.105963E+04	.231258E+04	-.149313E+05	-.694548E+03	-.157084E+05	-8.971	0.
ELEMENT NO. =		4						
1	-.140507E+05	-.635684E+03	.272355E+04	-.136079E+05	-.103829E+03	-.145826E+05	-11.050	0.
2	-.129503E+05	-.771685E+03	.251263E+04	-.139120E+05	-.273660E+03	-.134483E+05	-11.211	0.
3	-.162765E+05	-.534438E+03	.331560E+04	-.144035E+05	.135395E+03	-.169463E+05	-11.421	0.
4	-.105661E+05	-.107081E+04	.217283E+04	-.128314E+05	-.597214E+03	-.110397E+05	-12.296	0.
ELEMENT NO. =		5						
1	-.169374E+05	-.616907E+03	.343220E+04	-.150478E+05	.755061E+02	-.176298E+05	-11.406	0.

2	-.985701E+04	-.929878E+03	.189549E+04	-.123230E+05	-.544081E+03	-.102428E+05	-11.504	0.
3	-.158004E+05	-.584241E+03	.320362E+04	-.154607E+05	.627411E+02	-.164474E+05	-11.418	0.
4	-.109975E+05	-.917806E+03	.197734E+04	-.128391E+05	-.543789E+03	-.113715E+05	-10.711	0.
ELEMENT NO. =	6							
1	-.137487E+05	-.310136E+03	.270211E+04	-.153590E+05	.212832E+03	-.142716E+05	-10.954	0.
2	-.131377E+05	-.977521E+03	.241519E+04	-.139613E+05	-.515392E+03	-.135998E+05	-10.832	0.
3	-.100168E+05	-.385414E+03	.191198E+04	-.146078E+05	-.197402E+02	-.103824E+05	-10.827	0.
4	-.167785E+05	-.117197E+04	.363810E+04	-.159001E+05	-.365543E+03	-.175849E+05	-12.498	0.
ELEMENT NO. =	7							
1	-.722734E+04	.111484E+03	.143221E+04	-.134652E+05	.381084E+03	-.749694E+04	-10.661	0.
2	-.192763E+05	-.178155E+04	.492010E+04	-.173694E+05	-.492791E+03	-.205651E+05	-14.678	0.
3	-.478449E+04	-.226073E+03	.140219E+04	-.117250E+05	.170709E+03	-.518127E+04	-15.800	0.
4	-.209663E+05	-.258450E+04	.619192E+04	-.181578E+05	-.693317E+03	-.228575E+05	-16.984	0.
ELEMENT NO. =	8							
1	-.437572E+04	-.157141E+03	.160307E+04	-.101811E+05	.382897E+03	-.491576E+04	-18.618	0.
2	-.205255E+05	-.290008E+04	.724227E+04	-.175949E+05	-.306030E+03	-.231196E+05	-19.707	0.
3	-.569987E+04	-.196356E+04	.272300E+04	-.856768E+04	-.529486E+03	-.713394E+04	-27.774	0.
4	-.178667E+05	-.303932E+04	.725350E+04	-.151639E+05	-.811257E+02	-.208249E+05	-22.187	0.
ELEMENT NO. =	9							
1	-.820383E+04	-.168373E+04	.434615E+04	-.704395E+04	.489173E+03	-.103767E+05	-26.563	0.
2	-.140072E+05	-.338917E+04	.689124E+04	-.122639E+05	.945125E+00	-.173973E+05	-26.195	0.
3	-.124091E+05	-.260385E+04	.807719E+04	-.598991E+04	.194216E+04	-.169551E+05	-29.372	0.
4	-.832578E+04	-.512213E+04	.347770E+04	-.822056E+04	-.289508E+04	-.105528E+05	-32.635	0.
ELEMENT NO. =	10							
1	-.175228E+05	-.441212E+04	.989584E+04	-.667527E+04	.902688E+03	-.228376E+05	-28.239	0.
2	-.153824E+04	-.224010E+04	.286990E+04	-.322046E+04	.100211E+04	-.478044E+04	41.514	0.
3	-.253810E+05	-.152556E+05	.109737E+05	-.119169E+05	-.823306E+04	-.324035E+05	-32.617	0.
4	.708117E+04	.431688E+04	.684627E+03	.310099E+04	.724144E+04	.415661E+04	13.175	0.
	.	.	.	.	.	.		
	.	.	.	.	.	.		
	.	.	.	.	.	.		
	.	.	.	.	.	.		

etc.

AUTHOR INDEX

SUBJECT INDEX