Lecture 15

Sparse Recovery and ℓ_0 Sampling

Course: Algorithms for Big Data

Instructor: Hossein Jowhari

Department of Computer Science and Statistics
Faculty of Mathematics
K. N. Toosi University of Technology

Spring 2021

Sampling from streams: (Reservoir sampling)

Suppose we have a data stream $A = a_1, a_2, \ldots, a_m$. We want to sample a data item from the stream where

$$Pr(\text{ sampling } a_i) = \frac{1}{m}.$$

When the length of stream m is known: we just need to pick a random number $r \in [m]$ and then sample a_r.

When m is unknown: Let s denote the current sampled item. Before the stream $s = \varnothing$. For $i \geq 1$, we replace a_i with the current sample s with probability $\frac{1}{i}$, otherwise we keep the current sample.

$$Pr(\text{ sampling } a_i) = \frac{1}{i} \times \frac{i}{i+1} \times \frac{i+1}{i+2} \times \ldots \times \frac{m-1}{m} = \frac{1}{m}.$$

ℓ_1 sampling: positive updates

We have an initially zero vector $\boldsymbol{x} \in \mathbb{R}^n$. We receive a stream of positive updates to the coordinates of $\boldsymbol{x}$.

After the stream, we would like to have a sample coordinate $i \in [n]$ where

$$Pr(\text{ sampling } i) = \frac{x_i}{\sum_{j=1}^{n} x_j} = \frac{x_i}{\|\boldsymbol{x}\|_1}$$

$Pr(\text{ sampling coordinate } 1) = \frac{5}{20}$

$Pr(\text{ sampling coordiante } 2) = 0$

$Pr(\text{ sampling coordinate } 3) = \frac{1}{20}$

...

ℓ_1 sampling: positive updates

Set the current sample coordinate $t \leftarrow \varnothing$.
Set the current sum $s = 0$.

Given an update (i, u):
Add u to s. With probability $\frac{u}{s}$ set $t \leftarrow i$.

At the end of stream, return t as the sampled coordinate.

Analysis: To make the analysis easier, we can assume every coordinate is updated once. We are allowed to make this assumption because we can relabel the coordinate numbers and assume the underlying vector has a larger dimension. Consider the following example.

$$\ldots \quad (7_{(1)}, +3) \quad \ldots \quad (7_{(2)}, +9) \quad \ldots \quad (7_{(3)}, +6) \quad \ldots$$

$$Pr(i) = Pr(i_{(1)}) + Pr(i_{(2)}) + Pr(i_{(3)}) + \ldots$$

Now we can assume every coordinate is updated at most one time. Let $u_1, \ldots, u_m$ be the sequence of updates.

Suppose coordinate i is updated at position j in the stream. Let $s_j = u_1 + \ldots + u_j$.

$$Pr(i) = \frac{x_i}{s_j} \times \frac{s_j}{s_{j+1}} \times \ldots \frac{s_{m-1}}{s_m} = \frac{x_i}{s_m} = \frac{x_i}{\|\boldsymbol{x}\|_1}$$

What if we allow negative updates as well?

$(2, +3), (5, +4), (2, -1), (4, +9), (3, +1), (2, +3), (6, +2), (2, -4)$

Can we still pick a random coordinate (using little memory) without saving the entire vector?

This is a special case of a more general problem called ℓ_p sampling. There is a solution for it that uses $O(\log^2 n)$ bits of memory but we do not cover this in this course. See the references.

ℓ_0 sampling

Again assume we have an initially zero vector $\boldsymbol{x} \in \mathbb{R}^n$ that is changed by a stream of both positive and negative updates to its coordinates. At the end of the stream, we like to pick a non-zero coordinate in $\boldsymbol{x}$ uniformly at random (regardless of the weight of the coordinate). In other words,

$$Pr(\text{sampling non-zero coordinate } i) = \frac{1}{\|\boldsymbol{x}\|_0}$$

$Pr(\text{ sampling coordinate } 1) = \frac{1}{5}$

$Pr(\text{ sampling coordiante } 2) = 0$

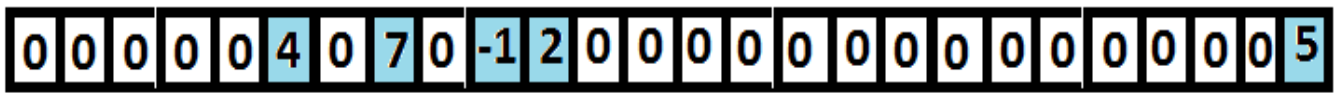
$Pr(\text{ sampling coordinate } 3) = \frac{1}{5}$

...

Sparse vectors

In order to describe the solution for ℓ_0 sampling we need to consider a variant of the sparse recovery problem.

Definition: A vector $\boldsymbol{x} \in \mathbb{R}^n$ with few non-zero entries is called a sparse vector. When $\boldsymbol{x}$ has at most k non-zero entries it is called k-sparse.



Number of non-zero coordinates in x is shown by $\|\boldsymbol{x}\|_0$.

A data stream problem: recovering a sparse vector

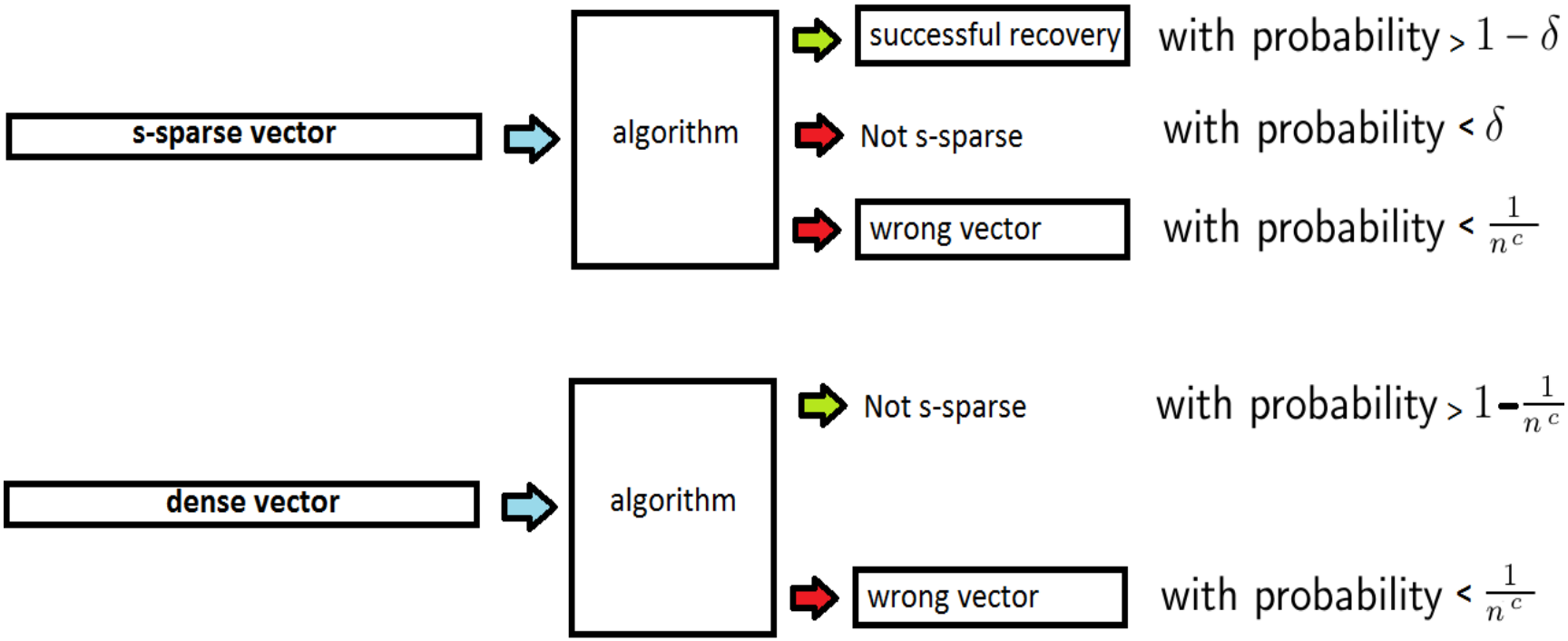
Suppose we have an initially zero vector $\boldsymbol{x} \in \mathbb{R}^n$. We get a stream of updates (addition and subtraction) to the coordinates of $\boldsymbol{x}$.

After the updates, we would like to answer the following questions: (Let $s \geq 0$ be a positive integer.)

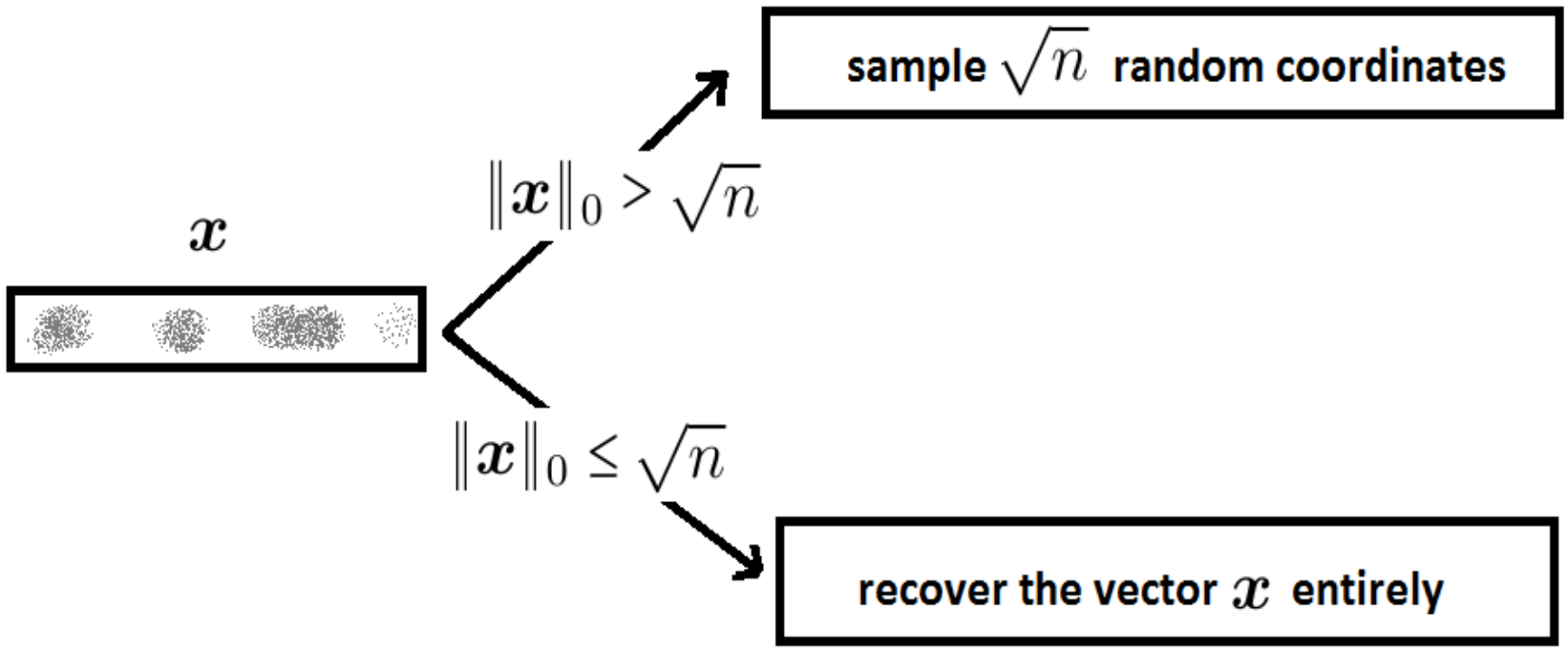
- Is the vector $\boldsymbol{x}$ s-sparse? In other words, is $\|\boldsymbol{x}\|_0 \leq s$?
- If $\boldsymbol{x}$ is s-sparse, recover $\boldsymbol{x}$ using small space.

A data stream problem: recovering a sparse vector

Theorem: There is a randomized algorithm that uses $O(s \log n \log(\frac{s}{\delta}))$ bits of space that recovers $\boldsymbol{x}$ exactly (in case if $\boldsymbol{x}$ is s-sparse) with probability $1 - \delta$. The algorithm might err and report that $\boldsymbol{x}$ is not s-sparse or recover a wrong vector $\boldsymbol{y} \neq \boldsymbol{x}$. The probability of recovering the wrong vector is $\frac{1}{n^{-c}}$ for some constant c.



This result immediately gives us a $O(\sqrt{n}\log^2 n)$ space solution for the ℓ_0 sampling problem.

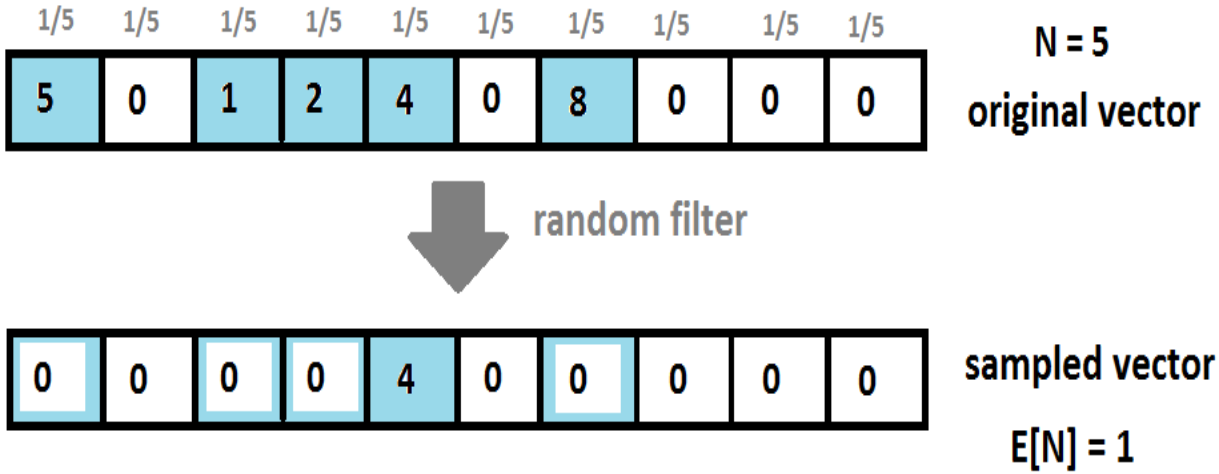


Note that the algorithm may return no sample at the end but this event happens with a small probability.

ℓ_0 sampling: a better idea

Suppose we know $N = \|\boldsymbol{x}\|_0$.

We sample each $i \in [n]$ with probability $\frac{1}{N}$ independently.



The sampled vector in expectation has $s = 1$ non-zero coordinates.

Applying the sparse recovery algorithm, we can recover the nonzero coordinates using $O(\log n \log \frac{1}{\delta})$ bits of space.

We have two obstacles in this approach:

- We do not know $N = \|\boldsymbol{x}\|_0$
- We need to store the random filter. In other words, we need to remember which coordinates are picked in the sampled vector.

We overcome both obstacles. To overcome the first obstacle, we guess the value of $\|\boldsymbol{x}\|_0$. We use multiple guesses (one of them will be good enough.)

To overcome the second obstacle, we use k-wise independent random numbers where $k = O(\log n)$. We need a few observations and lemmas on min-wise permutations (a topic not covered in this course.) See the references.

We try multiple guesses for $N = \|\boldsymbol{x}\|_0$. (for simplicity assume $n = 2^k$ for some k)

$$N = 1, \quad N = 2, \quad N = 4, \quad N = 8, \quad \ldots \quad N = \frac{n}{2}, \quad N = n$$

Assuming $\|\boldsymbol{x}\|_0 \geq 0$, consider $t \in \mathbb{Z}$ where $2^t \leq \|\boldsymbol{x}\|_0 \leq 2^{t+1}$.

$$N = 1, \quad N = 2, \quad N = 4, \quad \ldots \quad 2^t \quad 2^{t+1} \quad \ldots \quad N = \frac{n}{2}, \quad N = n$$

$\|\boldsymbol{x}\|_0$

If we sample the coordinates at a rate of $\frac{1}{2^t}$, in expectation there will be $O(1)$ non-zero coordinates in the sampled vector.

We try every guess $N = \{1, 2, 4, 8, \ldots, \frac{n}{2}, n\}$ in parallel.

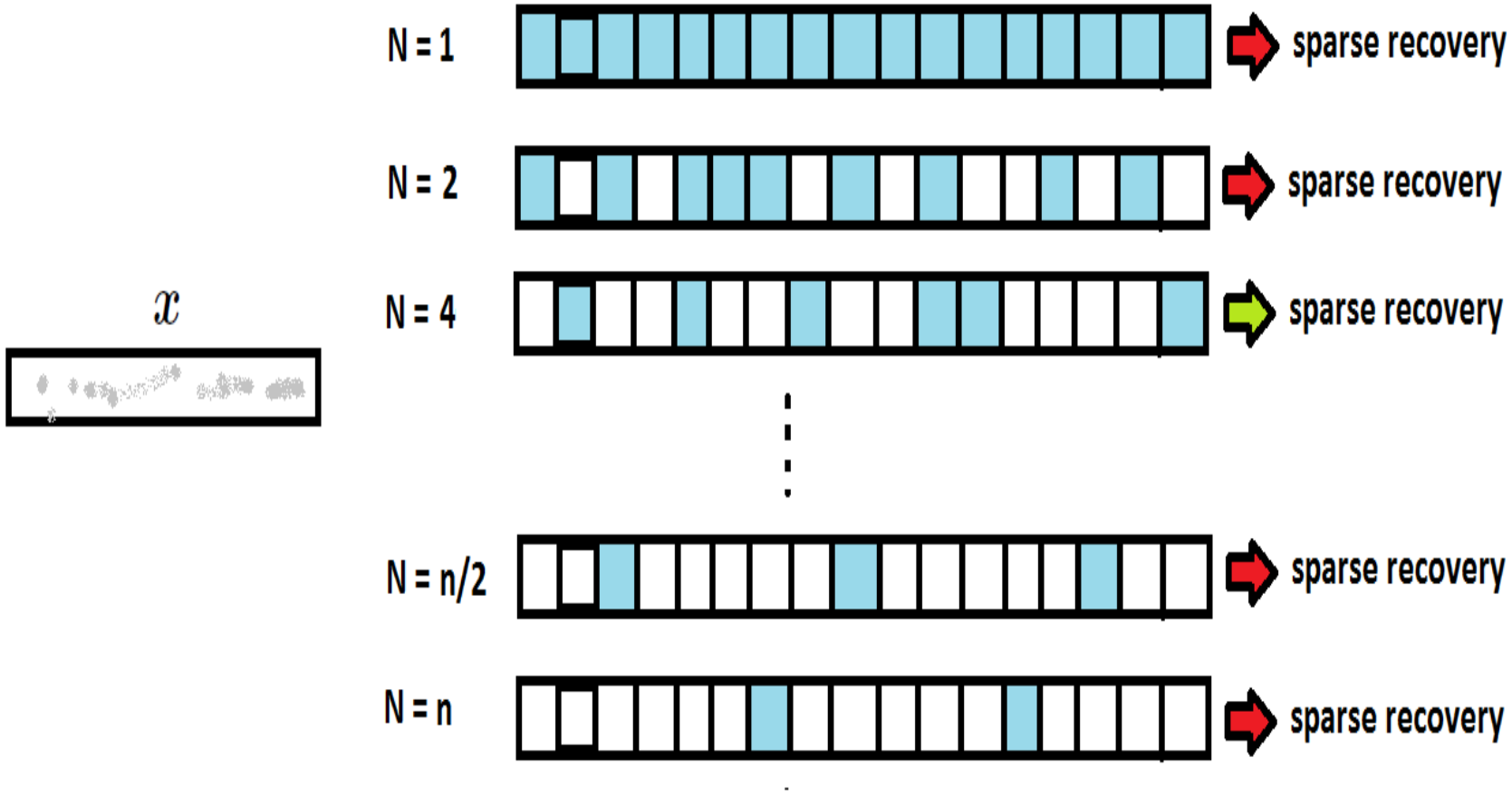
$$N = 1, \quad N = 2, \quad N = 4, \quad N = 8, \quad \ldots \quad N = \frac{n}{2}, \quad N = n$$

Assuming $\|\boldsymbol{x}\|_0 \geq 0$, consider $t \in \mathbb{Z}$ where $2^t \leq \|\boldsymbol{x}\|_0 \leq 2^{t+1}$.

$$N = 1, \quad N = 2, \quad N = 4, \quad \ldots \quad 2^t \quad 2^{t+1} \quad \ldots \quad N = \frac{n}{2}, \quad N = n$$

$\|\boldsymbol{x}\|_0$

If we sample the coordinates at a rate of $\frac{1}{2^t}$, in expectation there will be $O(1)$ non-zero coordinates in the sampled vector.



In each iteration, we assume number of non-zero coordinates in the sampled vector is $O(1)$. We run a sparse recovery procedure with parameter $s = O(1)$. This requires $O(\log n)$ bits of space for constant δ. One of the iterations will be successful. Total space complexity is $O(\log n \times \log n) = O(\log^2 n)$ bits of space.

There is a randomized algorithm that given a stream of positive and negative updates on a vector $\boldsymbol{x} \in \mathbb{R}^n$ with probability at least $3/4$ returns a random non-zero coordinate where each non-zero coordinate is returned with probability

$$\frac{1}{\|\boldsymbol{x}\|_0} \pm \frac{1}{n^c}$$

With probability at most $1/4$ the algorithm might return no sample. The algorithm uses $O(\log^2 n)$ bits of space.