Exponentially Improved Dimension Reduction in ℓ_1

Yi Li, David Woodruff, Taisuke Yasuda

July 20, 2021

Dimension Reduction

Avoiding the Curse of Dimensionality

- *Dimension Reduction*: techniques which reduce the dimensionality of datasets while (approximately) preserving properties of interest
    - Input: data in n-dimensions, where n is very large
    - Goal: want data to be in a much smaller r dimensions, for $r \ll n$
    - Want $f : \mathbb{R}^n \to \mathbb{R}^r$ such that $f(x)$ approximates x
- Motivation: *Curse of Dimensionality*
    - High dimensional spaces are often exponentially harder to work with

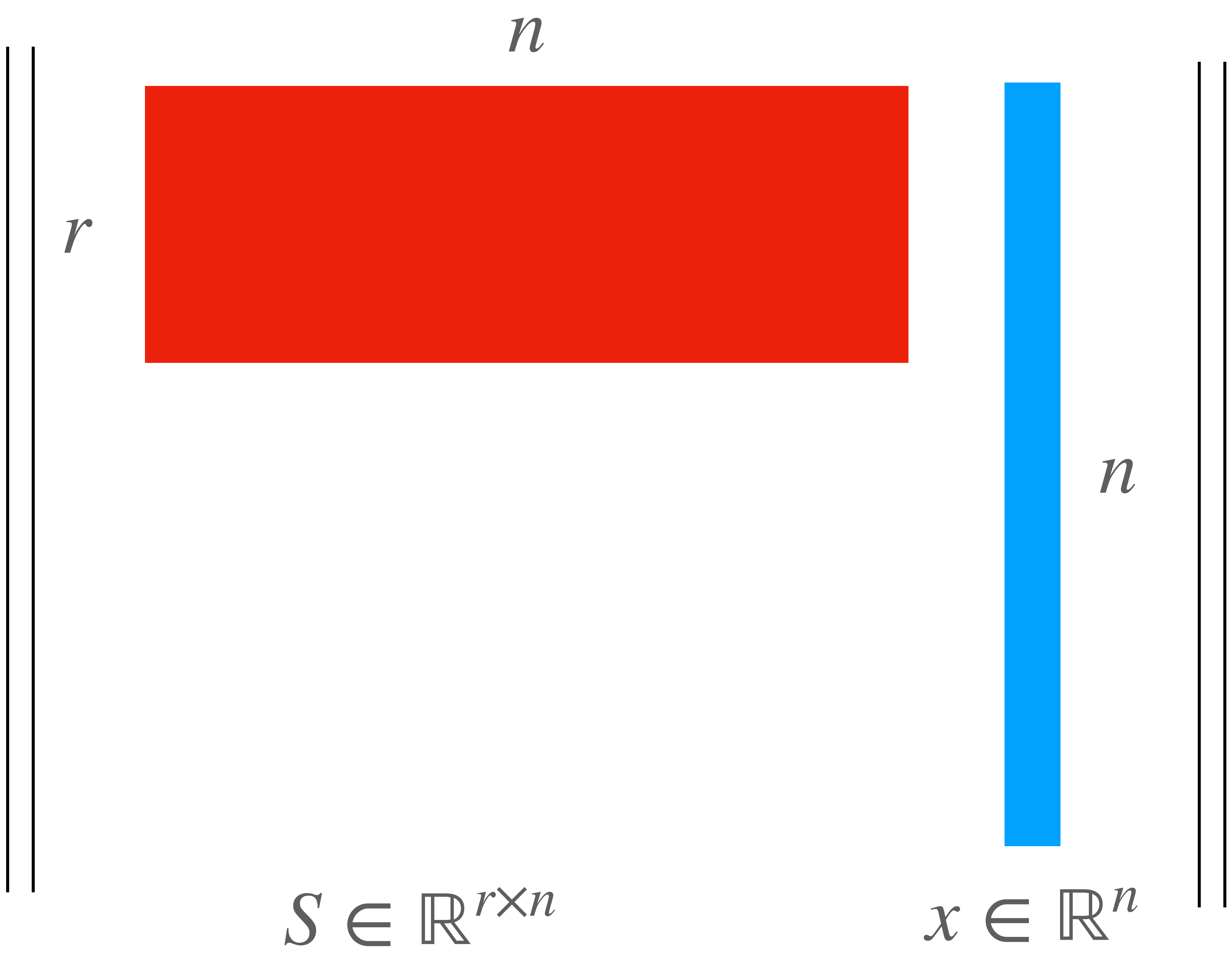
Dimension Reduction

Sketching

- *Sketching*: linear oblivious dimension reduction
  - Linear: $f : \mathbb{R}^n \to \mathbb{R}^r$ is $f(x) = Sx$ for a $r \times n$ matrix S
  - Oblivious: S is independent of the dataset

Dimension Reduction

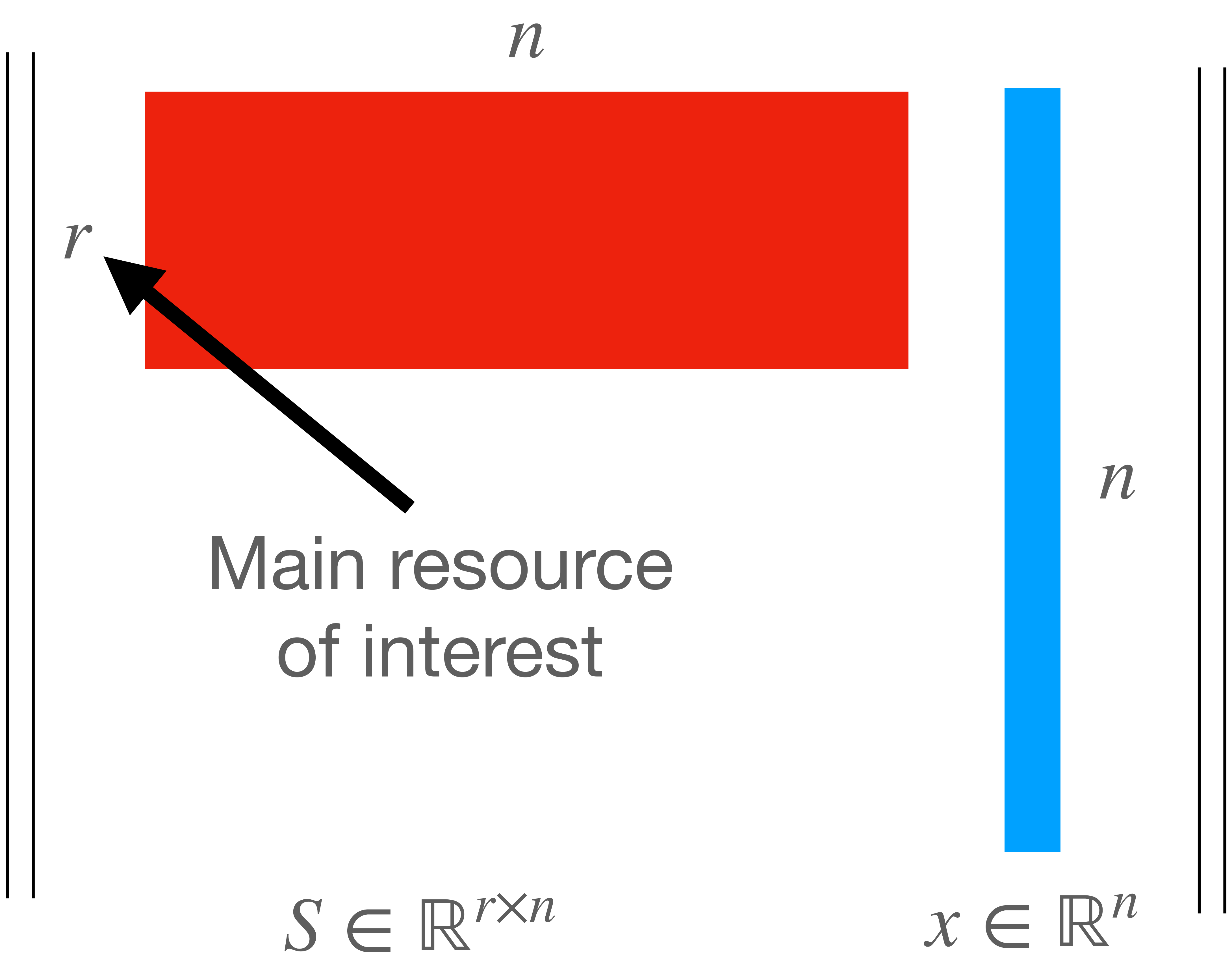
Sketching



$$(1-\varepsilon)\|x\| \leq \quad \|Sx\| \quad \leq (1+\varepsilon)\|x\|$$

Dimension Reduction

Sketching



$$||Sx|| = (1 \pm \varepsilon)||x||$$

with probability
at least 99%

Dimension Reduction

Sketching

- Why restrict ourselves to linear oblivious dimension reduction?
- Useful for:
    - Estimating pairwise distances

$$\|x - y\| \approx \|S(x - y)\| = \|Sx - Sy\|$$

    - Streaming environments: dynamic updates to x
        - Sketches are easy to update: $S(x + \Delta) = Sx + S\Delta$
    - Distributed environments: x and y belong to different servers
        - Sketches are easy to aggregate: $S(x + y) = Sx + Sy$

Dimension Reduction

Prior Work

- Johnson-Lindenstrauss (1984): dimension reduction for ℓ_2
  - Let S be an $r \times n$ matrix of i.i.d. Gaussian variables
    - Let $x \in \mathbb{R}^n$

      $r = \Theta(\varepsilon^{-2})$ $\quad\quad\|Sx\|_2 = (1 \pm \varepsilon)\|x\|_2$
    - Let $X \subseteq \mathbb{R}^n$ be a set of m vectors

      $r = \Theta(\varepsilon^{-2} \log m)$ $\quad\quad\|Sx\|_2 = (1 \pm \varepsilon)\|x\|_2$ for all $x \in X$
    - Let A be a $n \times d$ matrix

      $r = \Theta(\varepsilon^{-2} d)$ $\quad\quad\|Sx\|_2 = (1 \pm \varepsilon)\|x\|_2$ for all $x \in \text{span}(A)$

      "Subspace Embedding"

Dimension Reduction

Prior Work

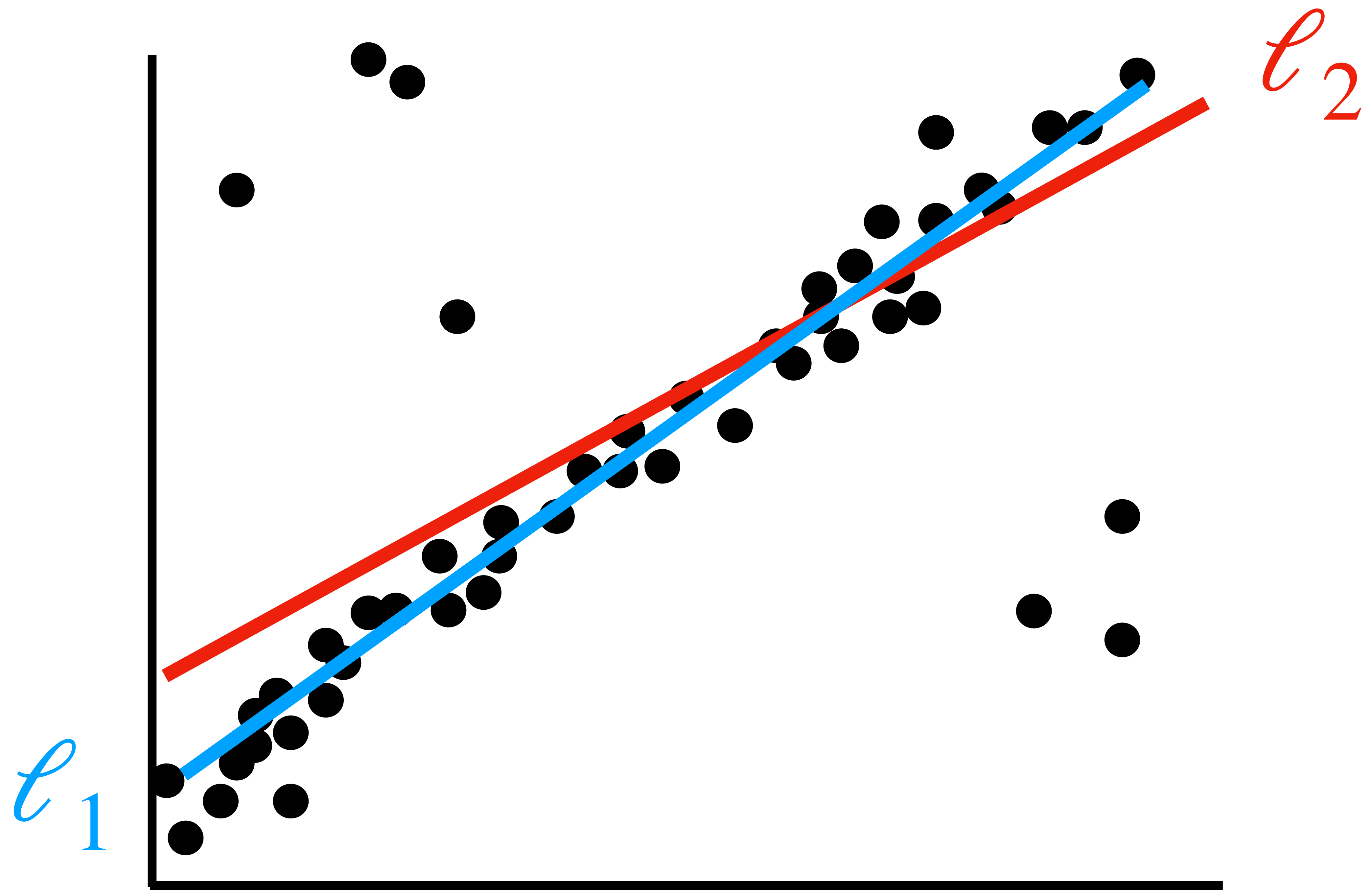
	ℓ_2 **Johnson-Lindenstrauss (1984)**
One Vector	ε^{-2}
m **Vectors**	$\varepsilon^{-2} \log m$
d**-dimensional Subspace**	$\varepsilon^{-2} d$

Applications

- Streaming algorithms
- Linear regression
- Low rank approximation
- Clustering
- Nearest neighbors

Dimension Reduction

What about for ℓ_1?



$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$

- a.k.a. Manhattan distance
- More robust than ℓ_2: more insensitive to outliers
  - E.g. least squares linear regression → least absolute deviations linear regression
  - (Kind of like mean → median)
- "The Manhattan distance is consistently more preferable than the Euclidean distance metric for high-dimensional data mining applications" [AHK01]

Dimension Reduction

What about for ℓ_1?

- Johnson-Lindenstrauss (1984):
    - S is a matrix of i.i.d. Gaussian variables → ℓ_2 dimension reduction
- Indyk (2006):
    - S is a matrix of i.i.d. Cauchy variables → ℓ_1 dimension reduction
        - … but with very low success probability
- Wang-Woodruff (2019):
    - S is a matrix of i.i.d. Cauchy variables → ℓ_1 dimension reduction
        - … but with doubly exponential bounds for r

Embeddings in ℓ_1

Prior Work

	ℓ_2 Johnson-Lindenstrauss (1984)	ℓ_1 Upper Bounds Wang-Woodruff (2019)	ℓ_1 Lower Bounds Wang-Woodruff (2019)
One Vector	ε^{-2}	$2^{2^{\varepsilon^{-2}}}$	
m **Vectors**	$\varepsilon^{-2}\log m$	$2^{2^{\varepsilon^{-2}\log m}}$	$2^{\sqrt{m}}$
d**-dimensional Subspace**	$\varepsilon^{-2}d$	$2^{2^{\varepsilon^{-2}d}}$	$2^{\sqrt{d}}$

*Suppresses big Oh and log factors

Embeddings in ℓ_1

Our Results

	ℓ_2 Johnson-Lindenstrauss (1984)	ℓ_1 Upper Bounds Li-Woodruff-Y (2021)	ℓ_1 Lower Bounds Wang-Woodruff (2019)
One Vector	ε^{-2}	$\cancel{2^{2^{\varepsilon^{-2}}}}$ $2^{\varepsilon^{-1}}$	
m **Vectors**	$\varepsilon^{-2}\log m$	$\cancel{2^{2^{\varepsilon^{-2}\log m}}}$ $2^{\varepsilon^{-1}m}$	$2^{\sqrt{m}}$
d**-dimensional Subspace**	$\varepsilon^{-2}d$	$\cancel{2^{2^{\varepsilon^{-2}d}}}$ $2^{\varepsilon^{-1}d}$	$2^{\sqrt{d}}$

*Suppresses big Oh and log factors

Embeddings in ℓ_1

Our Results

- Lowered dependence on d, ε^{-1} from doubly exponential to singly exponential
- Dependence on d is tight up to polynomial factors in the exponent
- ℓ_1 behaves very differently from ℓ_2
  - ℓ_1 doesn't care whether we embed d vectors or their entire span
  - For ℓ_2, there is an exponential difference

	ℓ_2	ℓ_1 **UB**	ℓ_1 **LB**
One Vector	ε^{-2}	$2^{\varepsilon^{-1}}$	
m **Vectors**	$\varepsilon^{-2} \log m$	$2^{\varepsilon^{-1} m}$	$2^{\sqrt{m}}$
d**-dimensional Subspace**	$\varepsilon^{-2} d$	$2^{\varepsilon^{-1} d}$	$2^{\sqrt{d}}$

Li-Woodruff-Y (2021)

Embeddings in ℓ_1

The Plan

- High level ideas used for our results
    - Embedding one vector
    - Embedding a subspace
- Proof of embedding one vector in detail

	ℓ_2	ℓ_1 UB	ℓ_1 LB
One Vector	ε^{-2}	$2^{\varepsilon^{-1}}$	
m **Vectors**	$\varepsilon^{-2}\log m$	$2^{\varepsilon^{-1}m}$	$2^{\sqrt{m}}$
d**-dimensional Subspace**	$\varepsilon^{-2}d$	$2^{\varepsilon^{-1}d}$	$2^{\sqrt{d}}$

Li-Woodruff-Y (2021)

Embeddings in ℓ_1

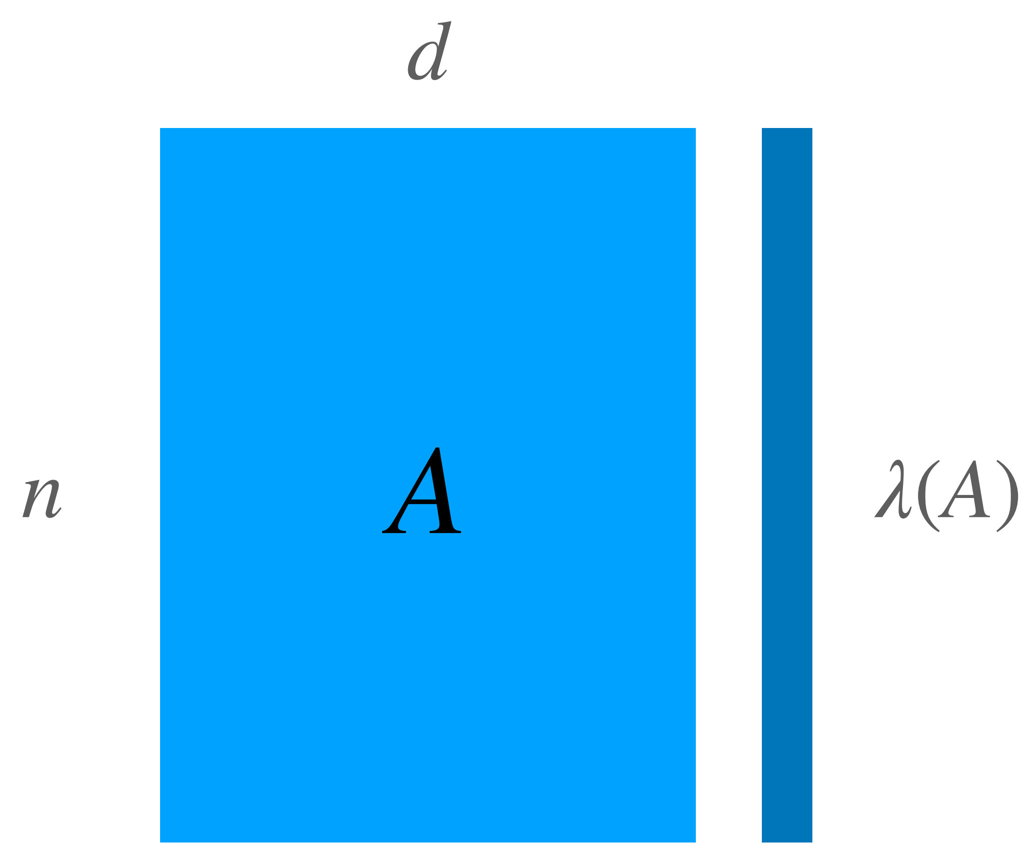
Ideas for Embedding One Vector

- Our starting point: M-sketch (Clarkson-Woodruff, 2015)
  - Classic techniques from the streaming literature: sampling and hashing
  - Achieves $O(1)$ distortion
- A new spin on this technique: randomized sampling rates
  - Achieves $(1+\varepsilon)$ distortion with singly exponential dependence on ε^{-1}

Embeddings in ℓ_1

Ideas for Embedding a Subspace

- Classic techniques for ℓ_2 rely on a net argument
    - Net: discretization of the unit sphere with $(1/\varepsilon)^d$ vectors
    - Apply the one vector embedding to every vector in the net
    - Still doubly exponential!
- Our idea: use ℓ_1 leverage scores (Clarkson—Drineas—Magdon-Ismail—Mahoney—Meng—Woodruff, 2013)



$\lambda(A)$ approximates the coordinates of $\dfrac{Av}{\|Av\|_1}$ for every $v \in \mathbb{R}^d$ up to a factor of d

- Apply one vector embedding to ℓ_1 leverage score vector with $(1 + \varepsilon/d)$ distortion

Embeddings in ℓ_1

The Plan

- High level ideas used for our results ✔
  - Embedding one vector ✔
  - Embedding a subspace ✔
- Proof of embedding one vector in detail
  - M-sketch (Clarkson—Woodruff, 2015): $O(1)$ distortion
  - Randomized sampling rates: $(1+\varepsilon)$ distortion

	ℓ_2	ℓ_1 **UB**	ℓ_1 **LB**
One Vector	ε^{-2}	$2^{\varepsilon^{-1}}$	
m **Vectors**	$\varepsilon^{-2}\log m$	$2^{\varepsilon^{-1}m}$	$2^{\sqrt{m}}$
d**-dimensional Subspace**	$\varepsilon^{-2}d$	$2^{\varepsilon^{-1}d}$	$2^{\sqrt{d}}$

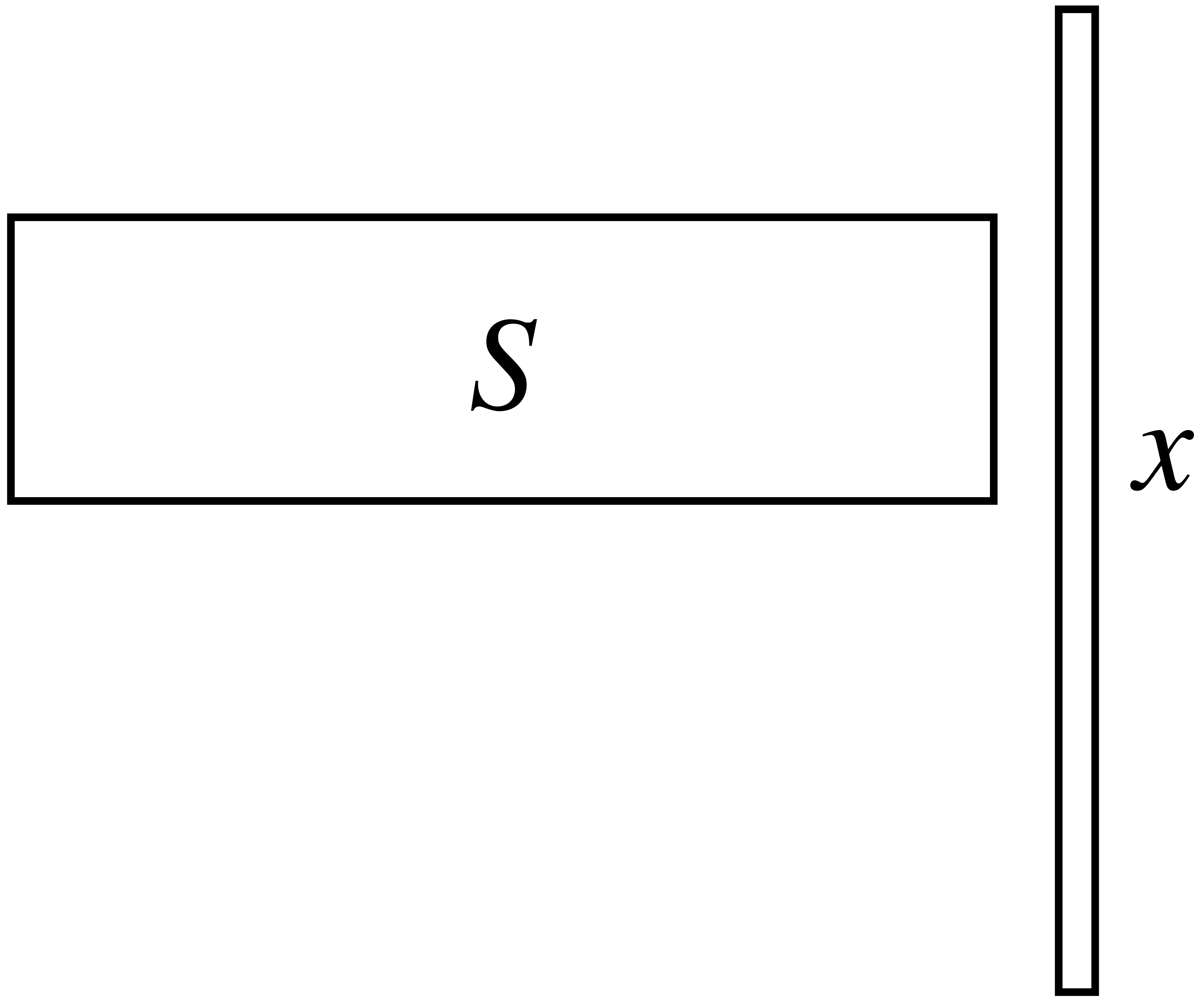
Li-Woodruff-Y (2021)

M-sketch: $O(1)$ distortion

M-sketch (Clarkson-Woodruff 2015)

Simplifying the Inputs

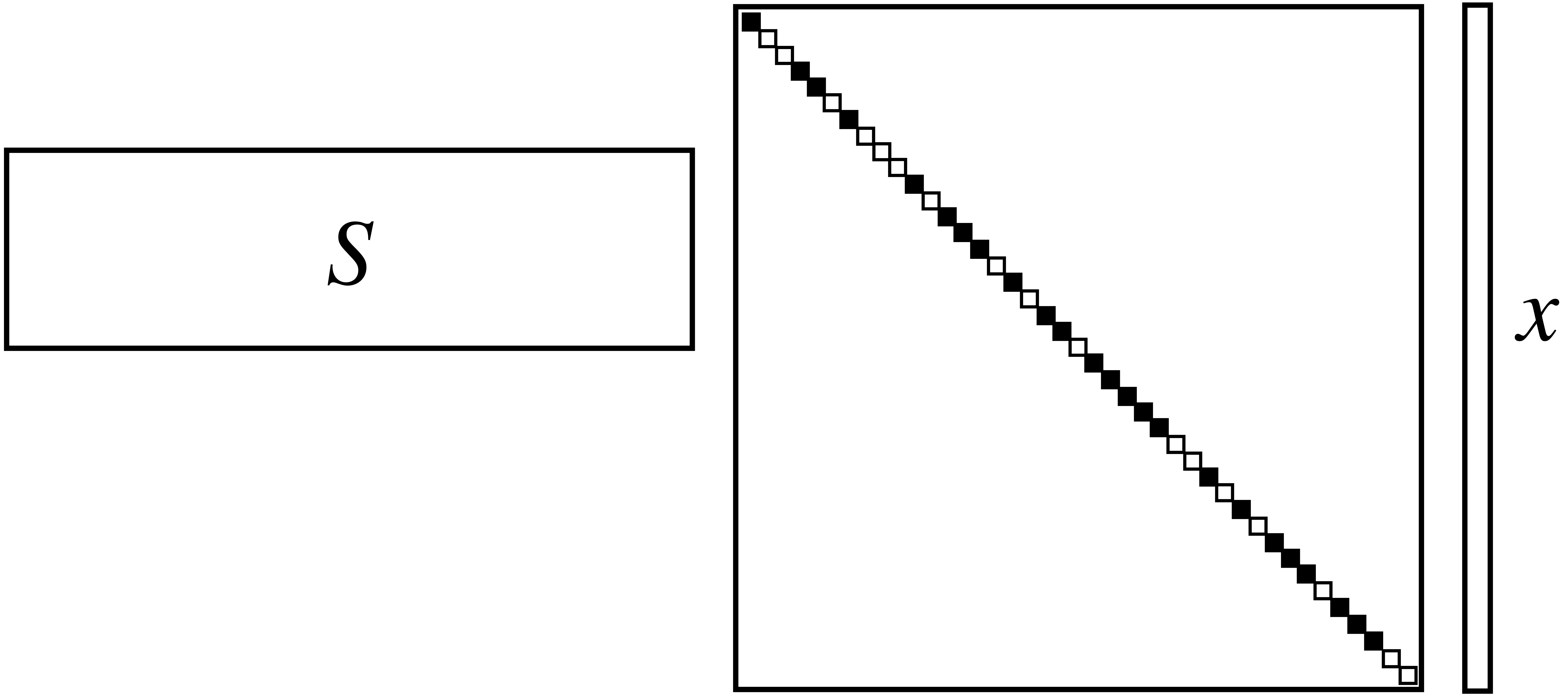
- Without loss of generality, the entries of $x \in \mathbb{R}^n$ have random signs



M-sketch (Clarkson-Woodruff 2015)

Simplifying the Inputs

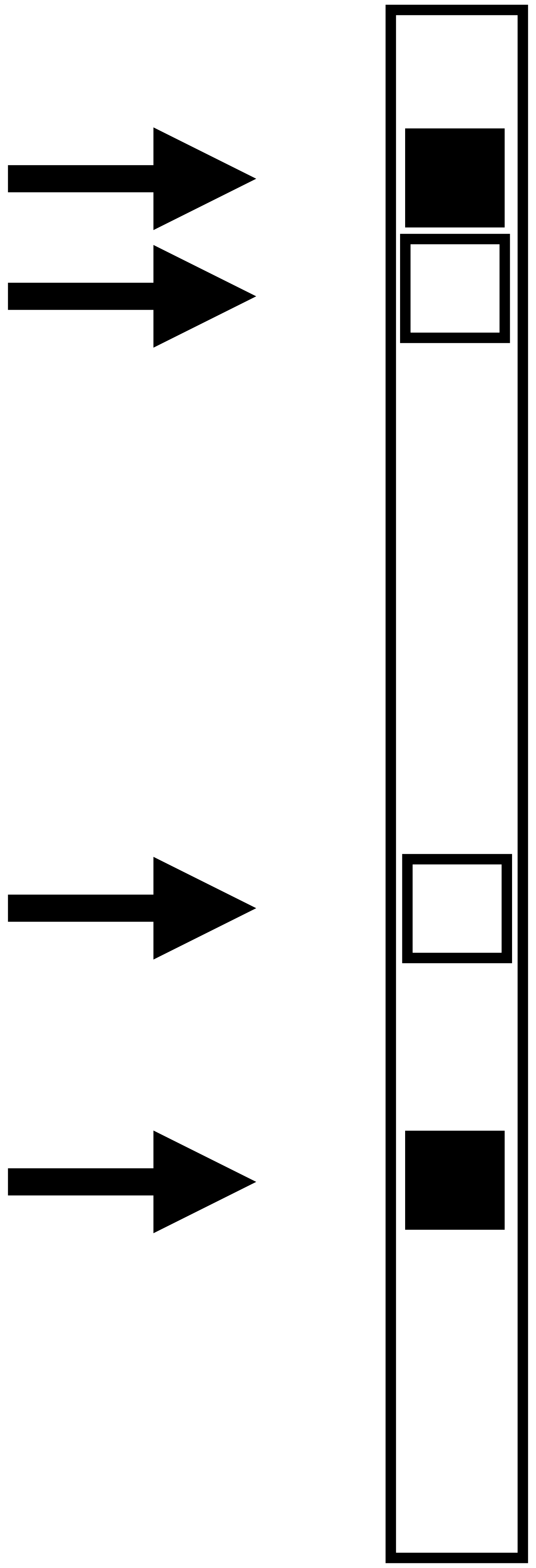
- Without loss of generality, the entries of $x \in \mathbb{R}^n$ have random signs



M-sketch (Clarkson-Woodruff 2015)

Simplifying the Inputs

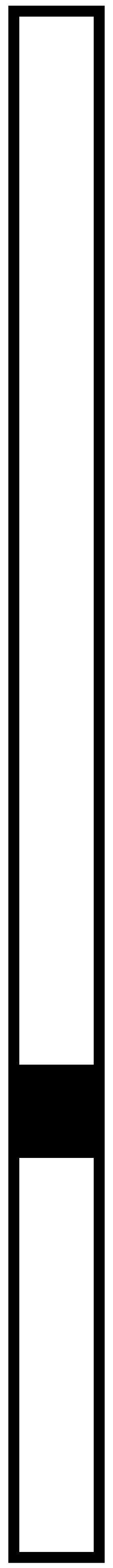
- Assume $x \in \mathbb{R}^n$ is an m-sparse vector of random signs
- Basically also without loss of generality



M-sketch (Clarkson-Woodruff 2015)

Case $m = 1$: The Ultimate Easy Case

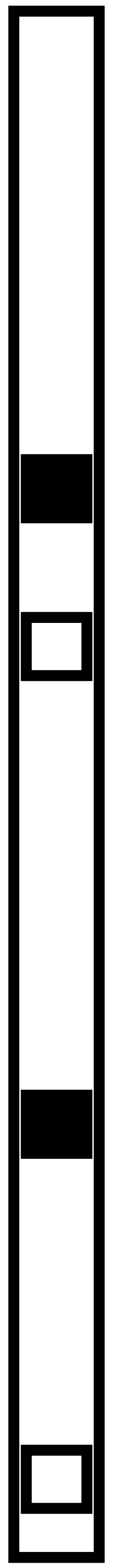
- Just add the entries of x!



M-sketch (Clarkson-Woodruff 2015)

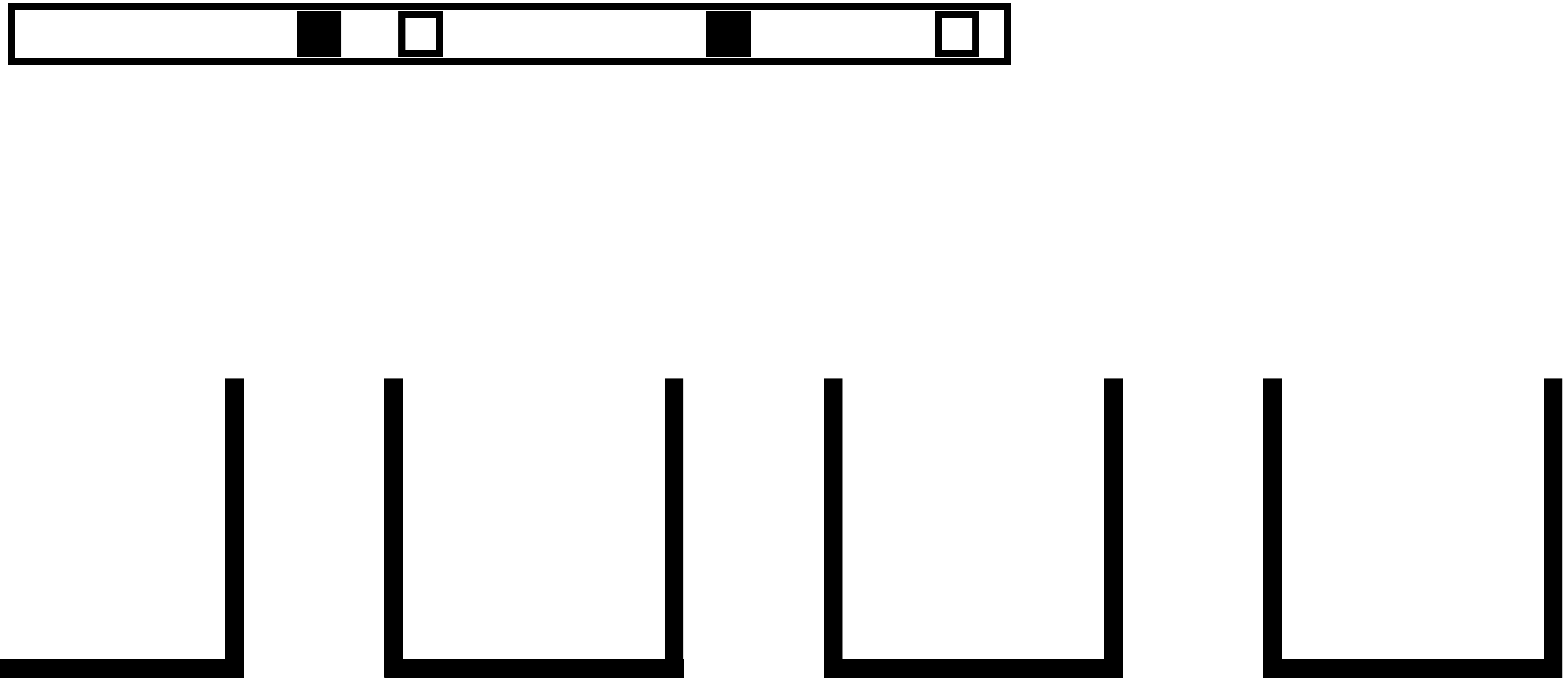
Case $m = \log n$

- Want to reduce to the $m = 1$ case
- Idea: hashing



M-sketch (Clarkson-Woodruff 2015)

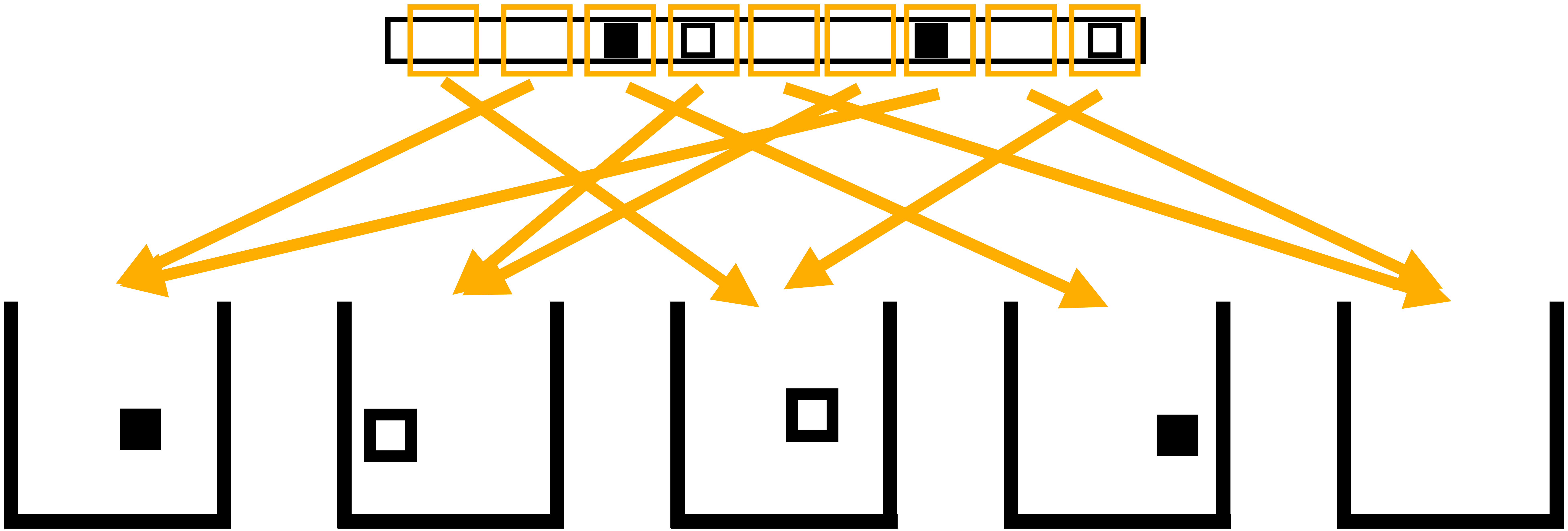
Case $m = \log n$: Hashing



$\log n$ buckets

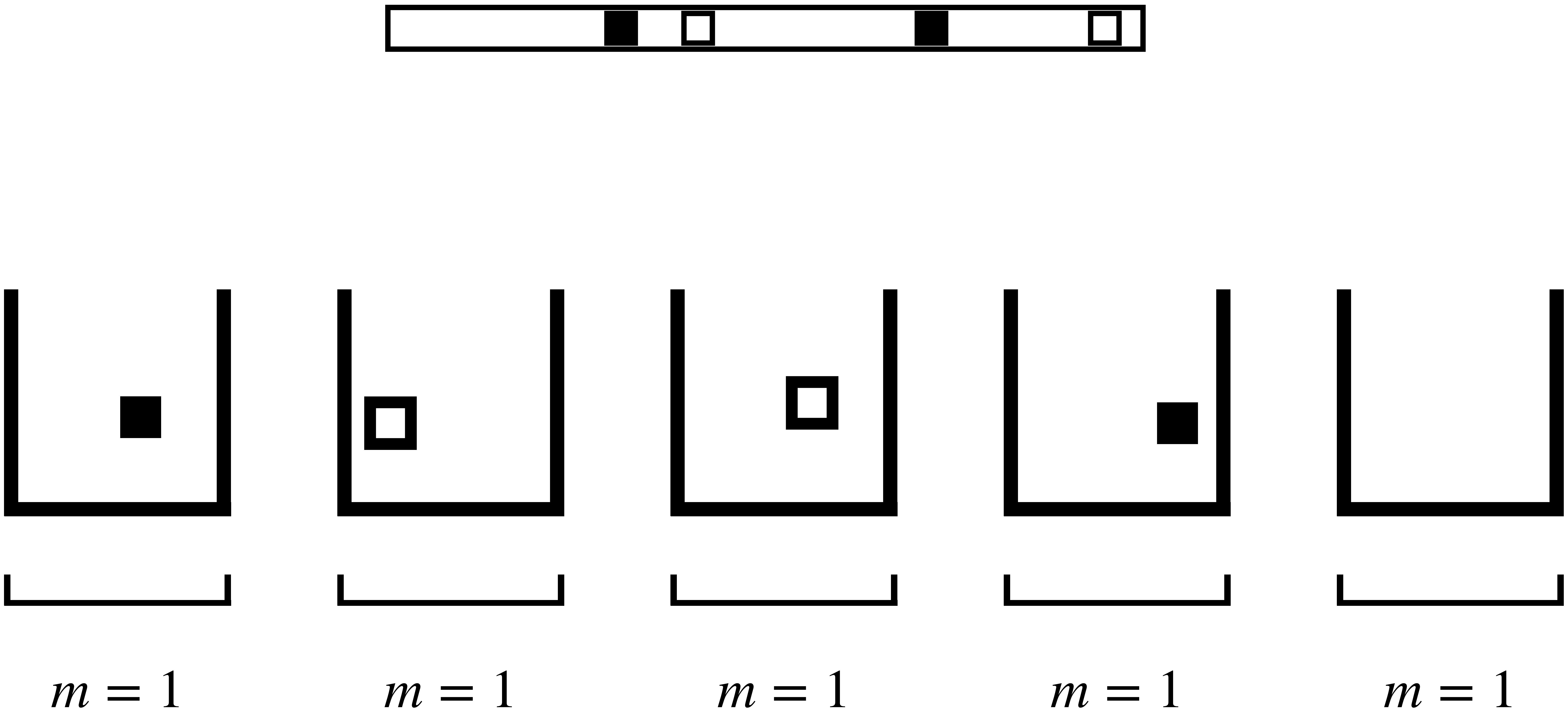
M-sketch (Clarkson-Woodruff 2015)

Case $m = \log n$: Hashing



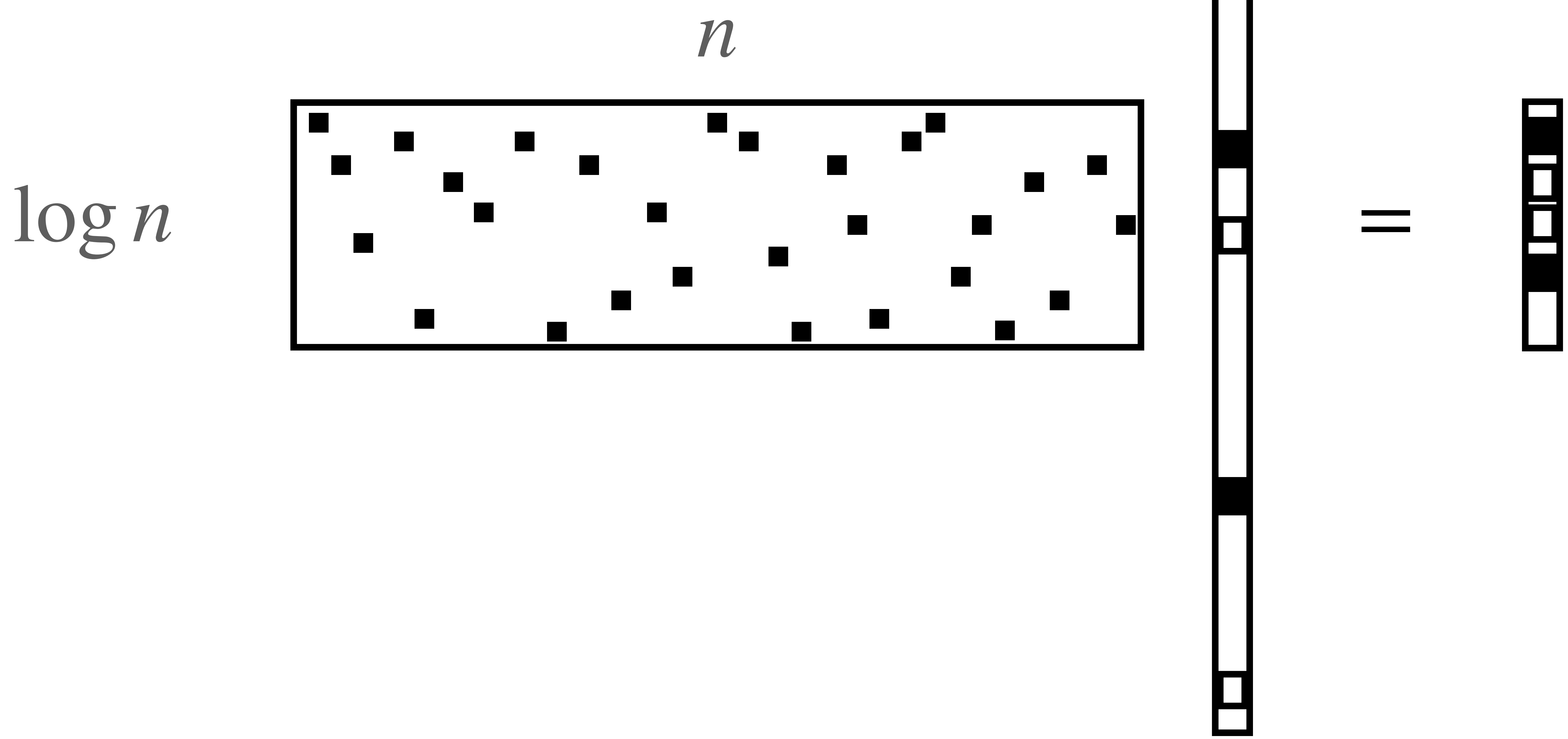
M-sketch (Clarkson-Woodruff 2015)

Case $m = \log n$: Hashing



M-sketch (Clarkson-Woodruff 2015)

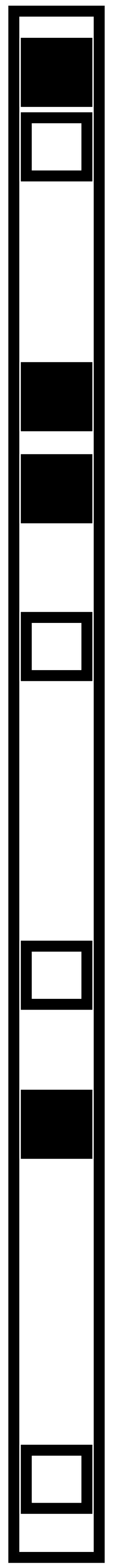
Case $m = \log n$: Hashing



M-sketch (Clarkson-Woodruff 2015)

Case $m = \log^2 n$

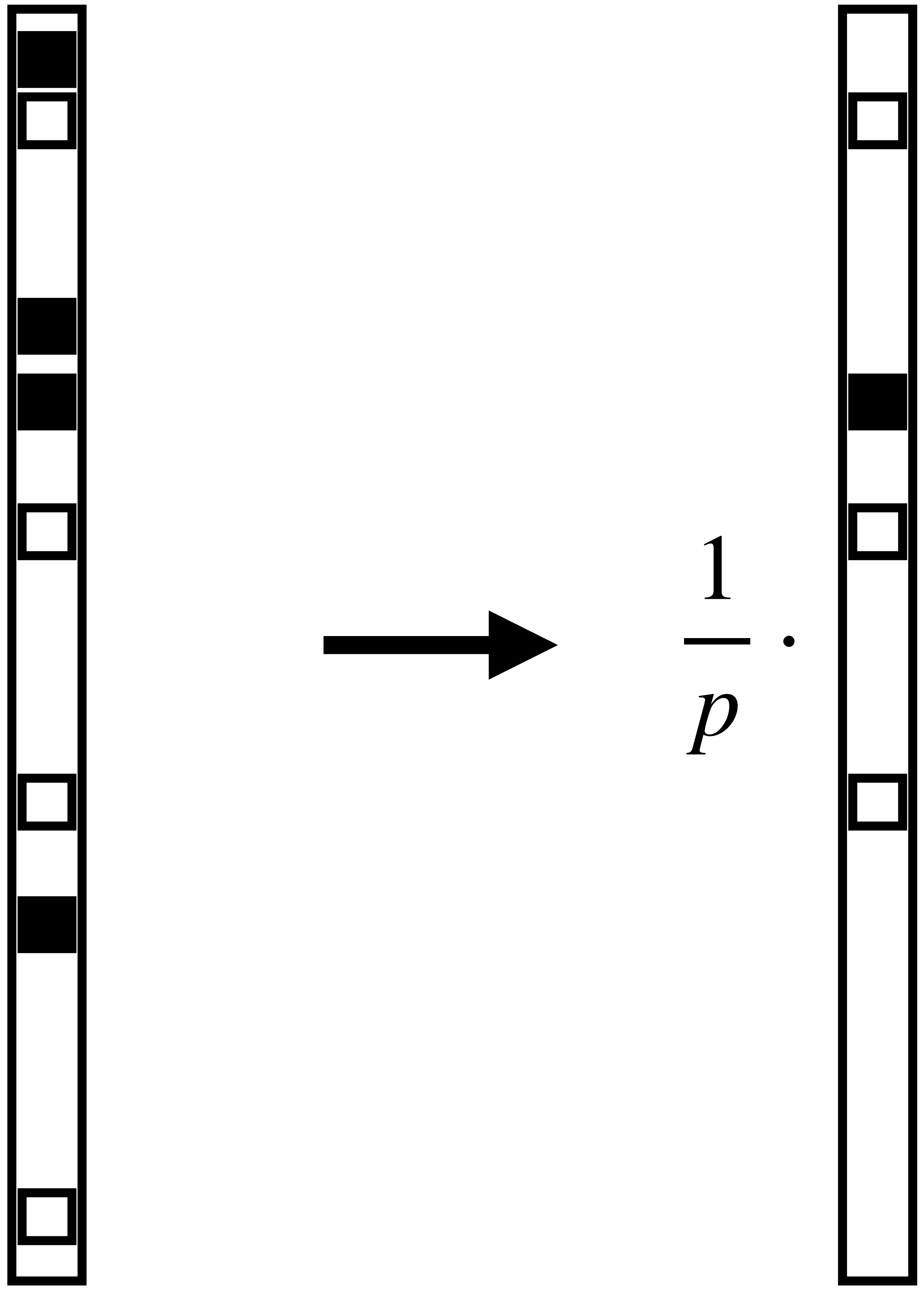
- With only $O(\log n)$ hash buckets, the buckets will be crowded…
- We could have more buckets, but we can't just keep doing that…
- Idea: sampling



M-sketch (Clarkson-Woodruff 2015)

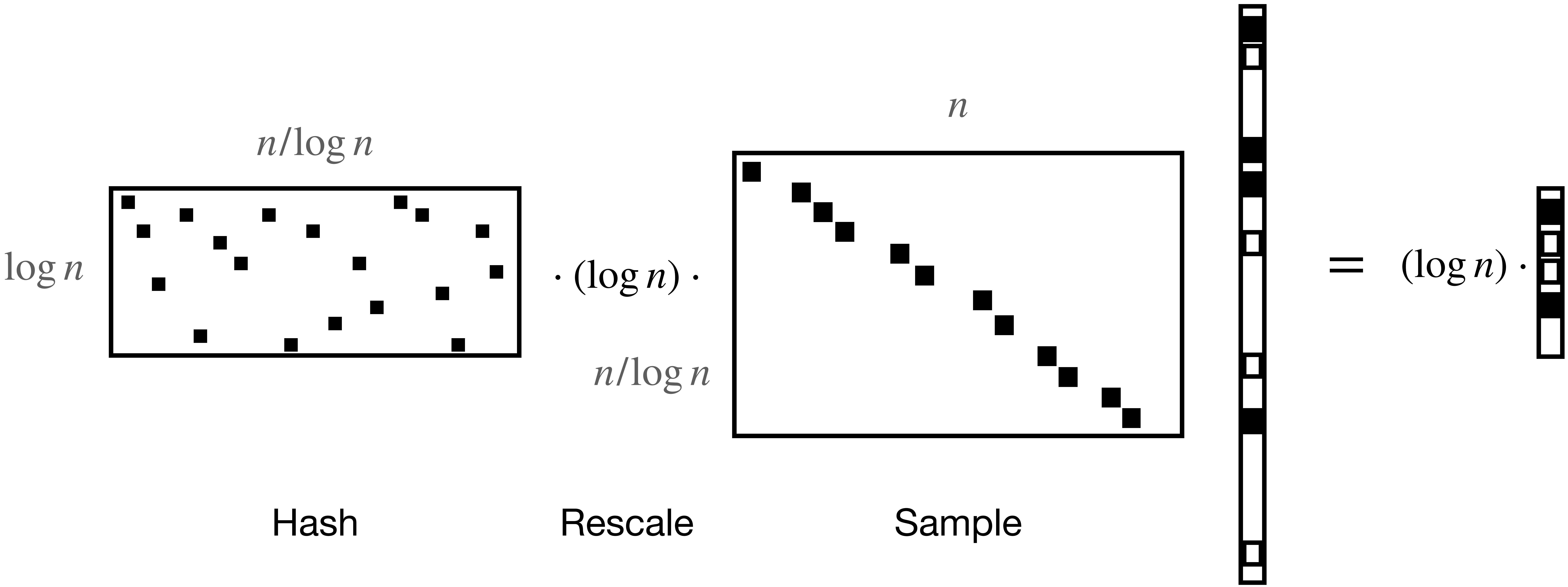
Case $m = \log^2 n$: Sampling

- Sample each coordinate with probability $p = (\log n)^{-1}$, then scale by p^{-1}
- Expected ℓ_1 norm is the same
- Only $p \cdot \log^2 n = \log n$ entries!



M-sketch (Clarkson-Woodruff 2015)

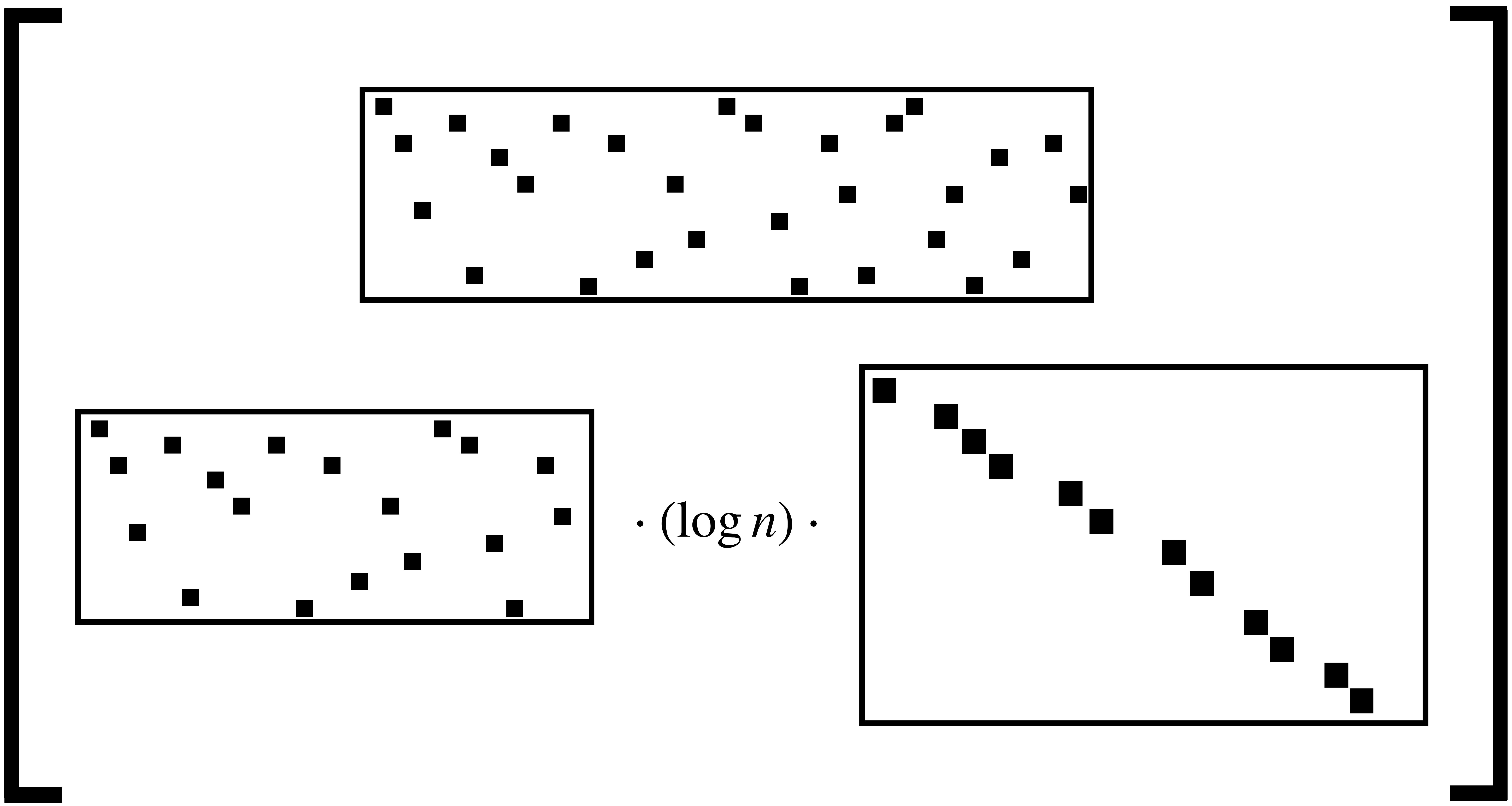
Case $m = \log^2 n$: Sampling



But what about the $m = \log n$ case?

M-sketch (Clarkson-Woodruff 2015)

Case $m = \log^2 n$: Sampling



In general…

M-sketch (Clarkson-Woodruff 2015)

General Case

- If $m = (\log n)^i$, then sample with probability $p_i = (\log n)^{-i+1}$ and hash into $O(\log n)$ buckets
- Stack all of these levels on top of each other

M-sketch (Clarkson-Woodruff 2015)

General Case

$O(\log n)$ blocks

Hash into $\log n$ buckets

⋮

Sample w.p. $p_i = (\log n)^{-i}$, rescale by $1/p_i$, and hash into $\log n$ buckets

⋮

Sample w.p. $p = 1/n$, rescale by n, and hash into $\log n$ buckets

M-sketch (Clarkson-Woodruff 2015)

Easy Analysis: $O(\log n)$ Distortion

$O(\log n)$ blocks

At most $\|x\|_1$ in expectation

⋮

At most $\|x\|_1$ in expectation

⋮

At most $\|x\|_1$ in expectation

At most $O(\log n)\|x\|_1$ in expectation all together

But do all $O(\log n)$ blocks contribute $\|x\|_1$?

M-sketch (Clarkson-Woodruff 2015)

Optimized Analysis: Crowded Hash Buckets

$O(\log n)$ blocks

At most $\|x\|_1$ in expectation

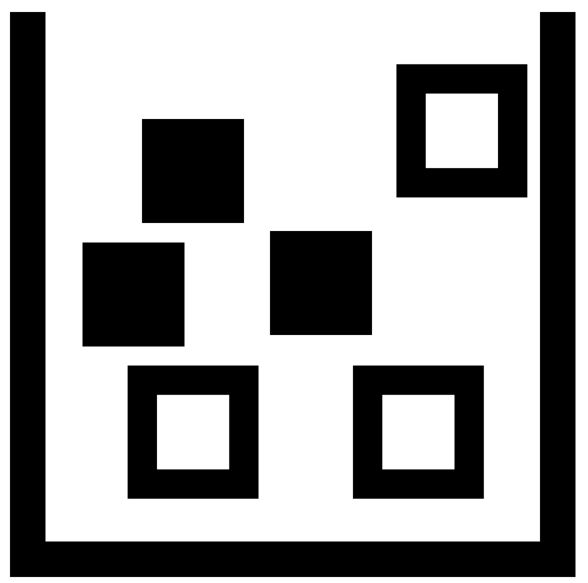
⋮

At most $\|x\|_1$ in expectation

⋮

At most $\|x\|_1$ in expectation

When the sampling rate is very high…



random signs cancel each other out!

M-sketch (Clarkson-Woodruff 2015)

Optimized Analysis: Low Sampling Rates

$O(\log n)$ blocks

$$\begin{bmatrix} \boxed{\text{At most } \|x\|_1 \text{ in expectation}} \\ \vdots \\ \boxed{\text{At most } \|x\|_1 \text{ in expectation}} \\ \vdots \\ \boxed{\boxed{\text{At most } \|x\|_1 \text{ in expectation}}} \end{bmatrix}$$

When the sampling rate is very low, no nonzero elements of x are sampled!

M-sketch (Clarkson-Woodruff 2015)

Optimized Analysis: $O(1)$ Distortion

$O(\log n)$ blocks

At most $\|x\|_1$ in expectation

⋮

At most $\|x\|_1$ in expectation

⋮

At most $\|x\|_1$ in expectation

Only one block whose sampling rate is "just right" will contribute an expected mass of $\|x\|_1$

$\Longrightarrow O(1)$ distortion

Randomized Sampling Rates: $(1 + \varepsilon)$ distortion

Randomized Sampling Rates

The Problem with M-sketch

- What do we need to do to get $(1+\varepsilon)$ distortion rather than $O(1)$?

M-sketch (Clarkson-Woodruff 2015)

Optimized Analysis: $O(1)$ Distortion

$O(\log n)$ blocks

At most $\|x\|_1$ in expectation

⋮

At most $\|x\|_1$ in expectation

⋮

At most $\|x\|_1$ in expectation

Only one block whose sampling rate is "just right" will contribute an expected mass of $\|x\|_1$

$\implies O(1)$ distortion

Randomized Sampling Rates

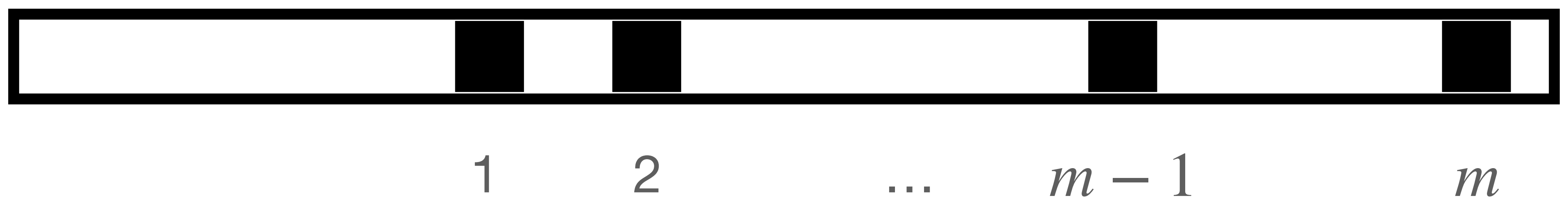
The Problem with M-sketch

- What do we need to do to get $(1+\varepsilon)$ distortion rather than $O(1)$?
  - Need to replace expected $\|x\|_1$ contribution with $(1 \pm \varepsilon)\|x\|_1$ with high probability
  - If the expected number of entries sampled is $pm \geq \varepsilon^{-2}$, then this is true by Chernoff bounds
  - For any fixed sampling rate p, a $\frac{1}{p}$-sparse vector is hard!
  - How to avoid this worst case? Randomize p!

Randomized Sampling Rates

Simplified Setting

- x is an m-sparse vector with 1s on its support



- We sample coordinates of x with probability $p_i = \dfrac{1}{B^i}$, and rescale by $\dfrac{1}{p_i}$

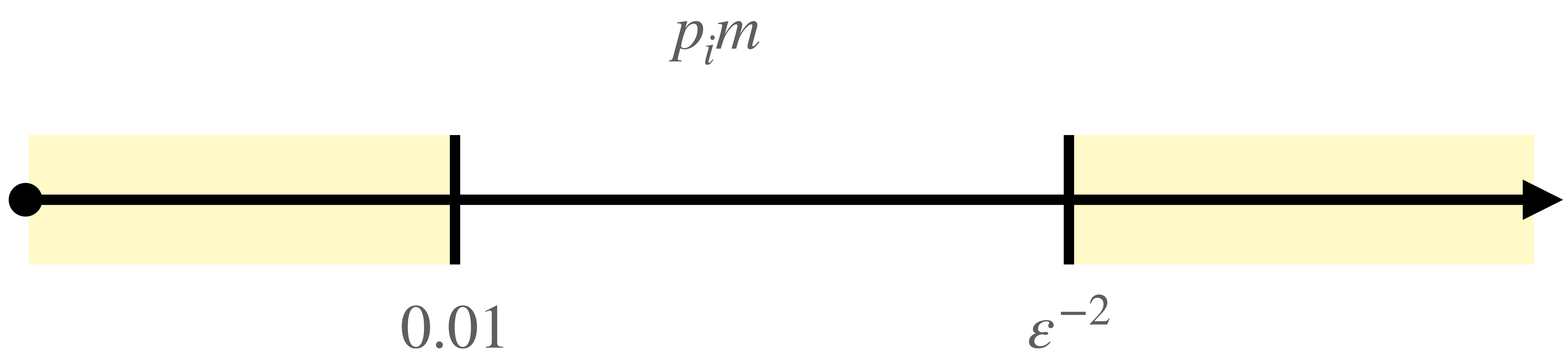
$\dfrac{1}{p_i} \cdot$ 

- With probability at least 99%, want to either:
  - Sample no entries, OR
  - Sample at least ε^{-2} entries

Randomized Sampling Rates

Simplified Analysis

- Casework on $p_i m$:



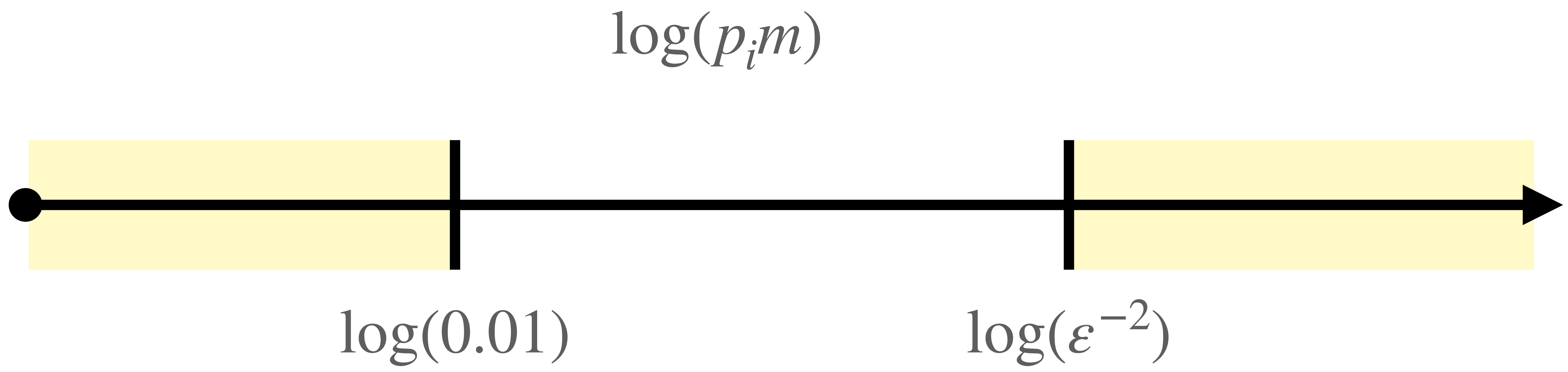
No entries sampled w.p. 99% by a union bound

Sampled mass concentrates around the expectation up to $(1 \pm \varepsilon)$ factor w.p. 99% by Chernoff bounds

Randomized Sampling Rates

Simplified Analysis

- Casework on $p_i m$:



No entries sampled w.p. 99% by a union bound

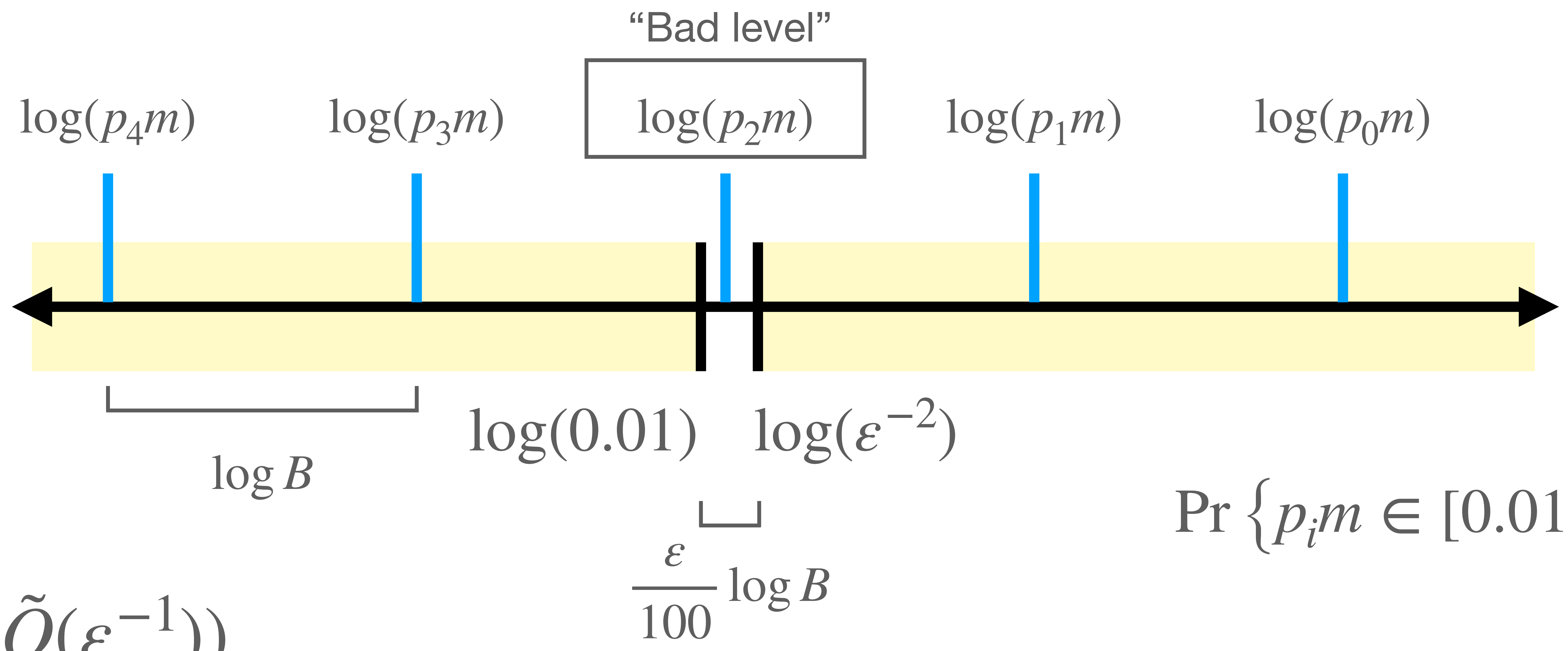
Sampled mass concentrates around the expectation up to $(1 \pm \varepsilon)$ factor w.p. 99% by Chernoff bounds

Randomized Sampling Rates

Simplified Analysis

- Casework on $p_i m$:

$$p_i = \frac{1}{B^i}$$



$$\Pr\left\{p_i m \in [0.01, \varepsilon^{-2}]\right\} \leq \frac{\varepsilon}{100}$$

$$B = \exp(\tilde{O}(\varepsilon^{-1}))$$

Randomized Sampling Rates

Finishing Up

- Let $B = \exp(\tilde{O}(\varepsilon^{-1}))$
- Let $U \sim [0,1]$ and $\tilde{B} = B^U$
- We now sample at rates $p_i = \dfrac{\tilde{B}}{B^i}$
- To accommodate for this, we need to change the number of hash buckets to exponential in ε^{-1} as well
- This works!

Conclusion

- Exponentially improved d, ε^{-1} bounds for dimension reduction in ℓ_1
- Dependence on d is tight up to polynomial factors in the exponent
- New techniques:
  - Randomized sampling rates: classic streaming techniques with a twist
  - Avoiding a net argument through the use of ℓ_1 leverage scores

	ℓ_2	ℓ_1 **UB**	ℓ_1 **LB**
One Vector	ε^{-2}	$2^{\varepsilon^{-1}}$	
m **Vectors**	$\varepsilon^{-2}\log m$	$2^{\varepsilon^{-1}m}$	$2^{\sqrt{m}}$
d**-dimensional Subspace**	$\varepsilon^{-2}d$	$2^{\varepsilon^{-1}d}$	$2^{\sqrt{d}}$

Li-Woodruff-Y (2021)