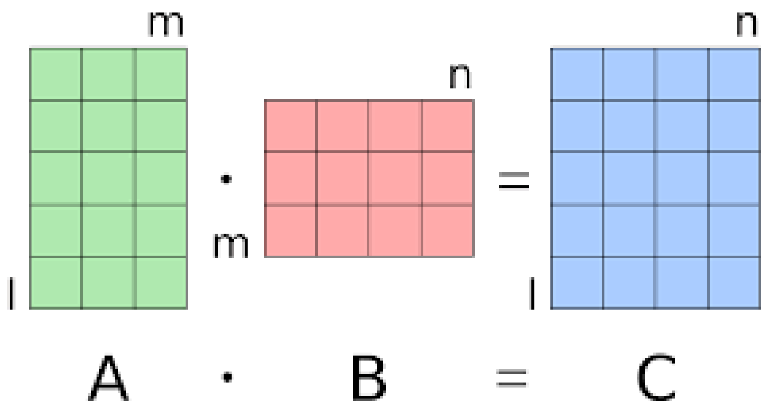
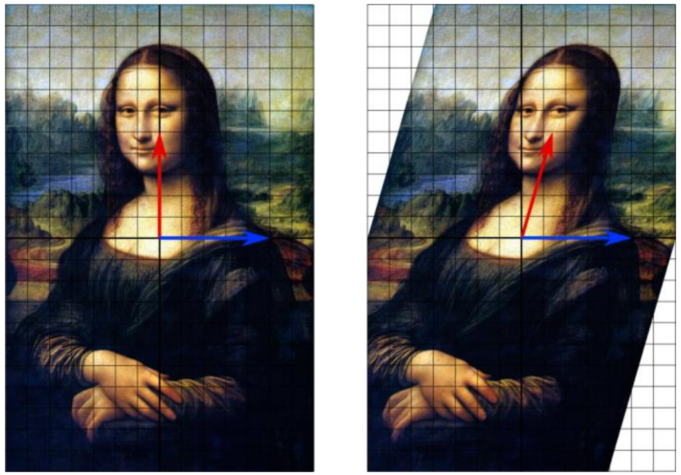
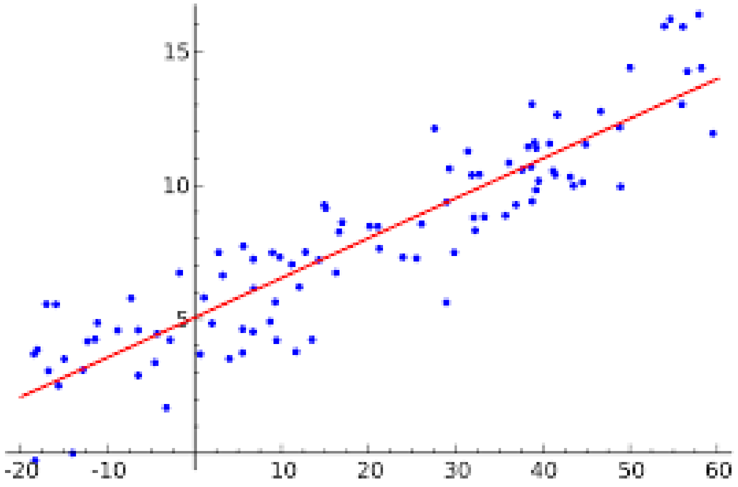
1 0 1 1 1 0 0 1 1 0 1

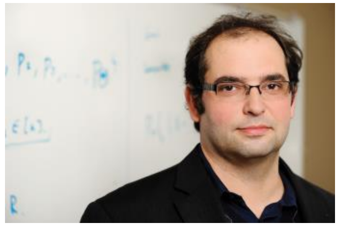
Numerical Linear Algebra in the Sliding Window Model





Wikipedia





Vladimir Braverman

Johns Hopkins University

Petros Drineas

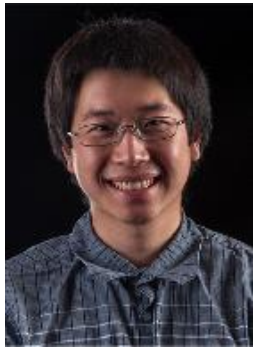
Purdue University

Jalaj Upadhyay

Johns Hopkins University

David P. Woodruff

Carnegie Mellon University



Samson Zhou

Indiana University

Streaming / Sliding Window Model

- Input: Elements of an underlying data set S, which arrives sequentially
- Output: Evaluation (or approximation) of a given function
- Goal: Use space *sublinear* in the size of the input S

Streaming / Sliding Window Model

- Input: Elements of an underlying data set S, which arrives sequentially
- Output: Evaluation (or approximation) of a given function in space *sublinear* in the size of the input
  - Quantiles, heavy-hitters, norm estimation, distinct elements, sampling
  - Matchings, number of triangles, spanners, sparsifiers
  - Numerical linear algebra (matrix multiplication, spectral approximation,…)
  - Minimum enclosing ball, Clustering (k-means, k-median, k-centers,…)
  - Submodular optimization
  - Strings (pattern matching, periodicity, edit distance, Parikh matching,…)
  - Codeword testing

Streaming / Sliding Window Model

- Input: Elements of an underlying data set S, which arrives sequentially
- Output: Evaluation (or approximation) of a given function
- Goal: Use space *sublinear* in the size of the input S

- Sliding Window: "Only the W most recent updates form the underlying data set S"
  - Recent interactions, time sensitive

$\boxed{1\ 0\ 1\ 1\ 1\ 0\ 0\ 1}$

Streaming / Sliding Window Model

- **Input**: Elements of an underlying data set S, which arrives sequentially
- **Output**: Evaluation (or approximation) of a given function
- **Goal**: Use space *sublinear* in the size of the input S

- **Sliding Window**: "Only the W most recent updates form the underlying data set S"
  - Recent interactions, time sensitive

$$1\ \boxed{0\ 1\ 1\ 1\ 0\ 0\ 1\ 1}$$

Streaming / Sliding Window Model

- Input: Elements of an underlying data set S, which arrives sequentially
- Output: Evaluation (or approximation) of a given function
- Goal: Use space *sublinear* in the size of the input S

- Sliding Window: "Only the W most recent updates form the underlying data set S"
  - Recent interactions, time sensitive

$$1\ 0\ \boxed{1\ 1\ 1\ 0\ 0\ 1\ 1\ 0}$$

Streaming / Sliding Window Model

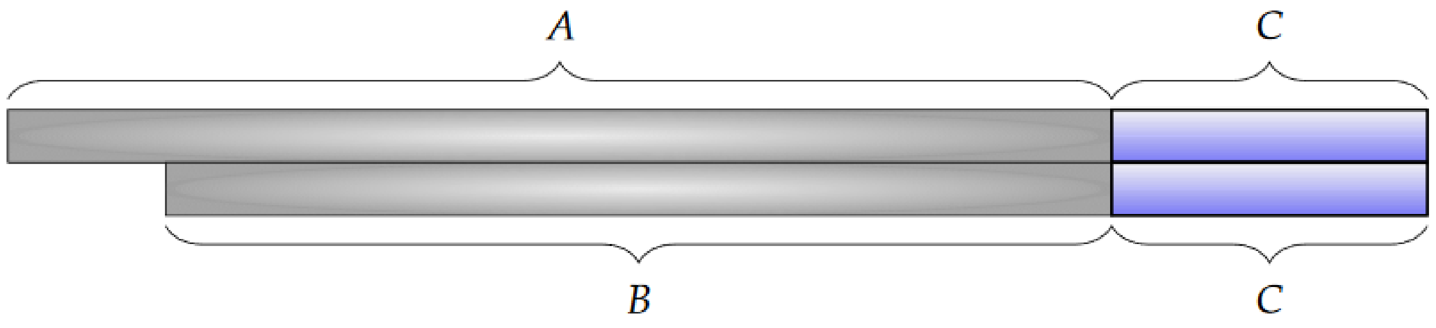
- Input: Elements of an underlying data set S, which arrives sequentially
- Output: Evaluation (or approximation) of a given function
- Goal: Use space *sublinear* in the size of the input S

- Sliding Window: "Only the W most recent updates form the underlying data set S"
  - Recent interactions, time sensitive

$1\ 0\ 1\ \boxed{1\ 1\ 0\ 0\ 1\ 1\ 0\ 1}$

Sliding Window Algorithms

- Suppose we are trying to approximate some given function
  1. Suppose we have a streaming algorithm for this function
  2. Suppose this function is "smooth": If $f(B)$ is a "good" approximation to $f(A)$, then $f(B \cup C)$ will always be a "good" approximation to $f(A \cup C)$.



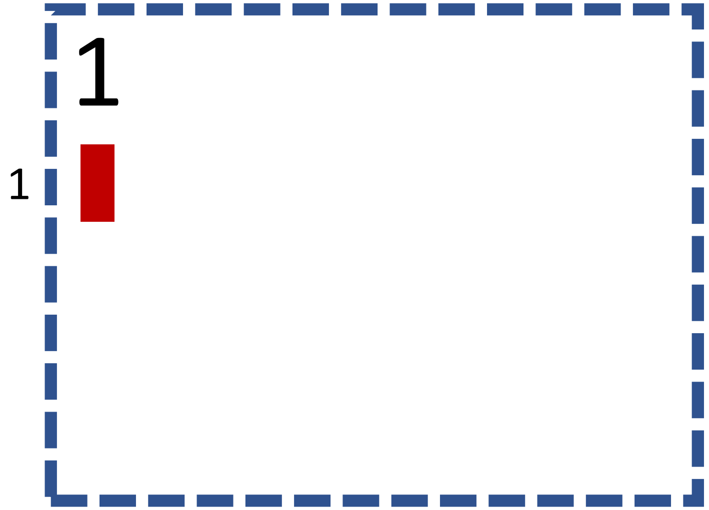
- Smooth histogram framework [BO07] gives a sliding window algorithm for this function

Smooth Histogram

- ❖ Suppose we are trying to approximate some given function
- ❖ Smooth histogram framework [BO07] gives a sliding window algorithm for this function
- ❖ Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- ❖ Each time there are three instances that report "close" values, delete the middle one
- ❖ Use a number of different starting points to "sandwich" the sliding window

Smooth Histogram

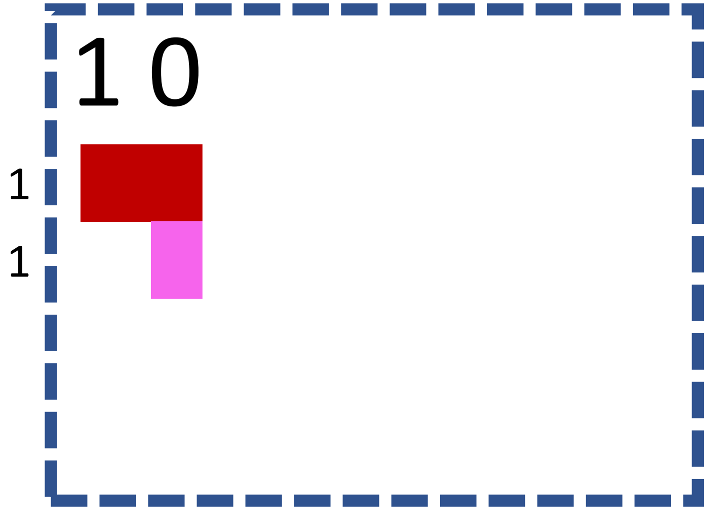
- ❖ Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- ❖ Each time there are three instances that report “close” values, delete the middle one
- ❖ Use a number of different starting points to “sandwich” the sliding window



- ❖ Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

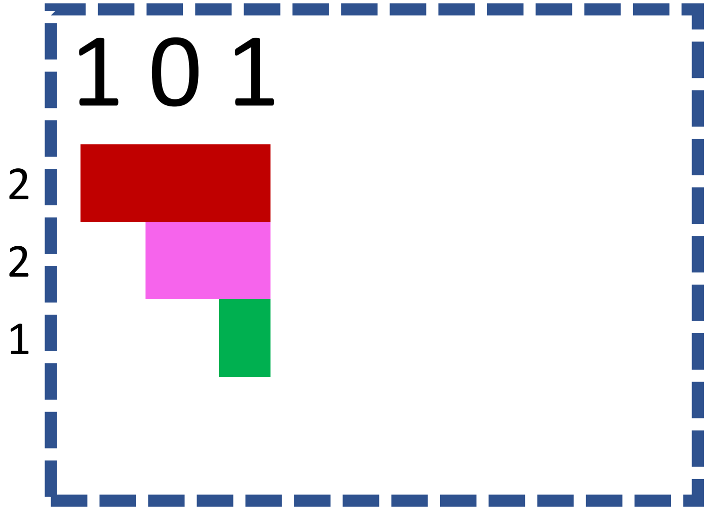
- Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- Each time there are three instances that report “close” values, delete the middle one
- Use a number of different starting points to “sandwich” the sliding window



- Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

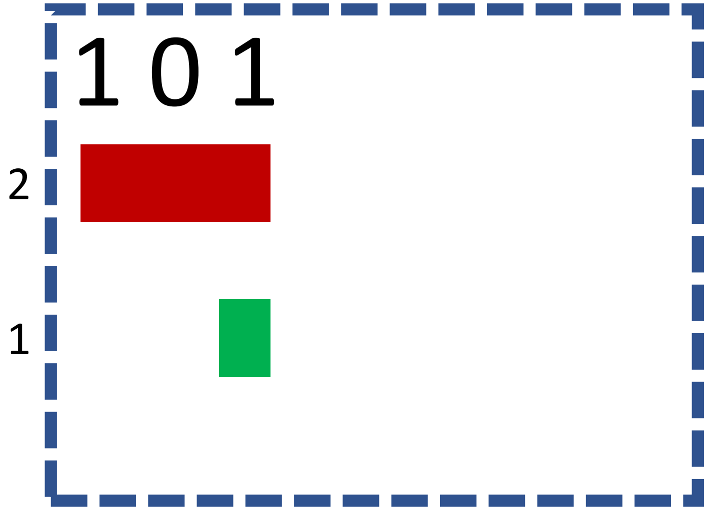
- Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- Each time there are three instances that report “close” values, delete the middle one
- Use a number of different starting points to “sandwich” the sliding window



- Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

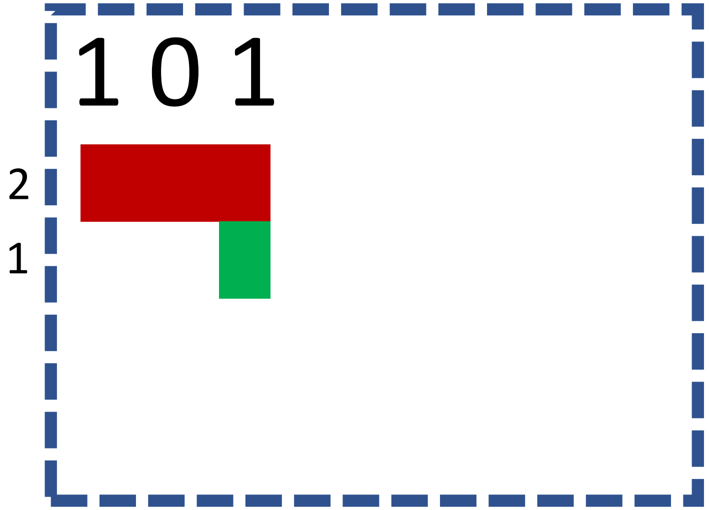
- Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- Each time there are three instances that report “close” values, delete the middle one
- Use a number of different starting points to “sandwich” the sliding window



- Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

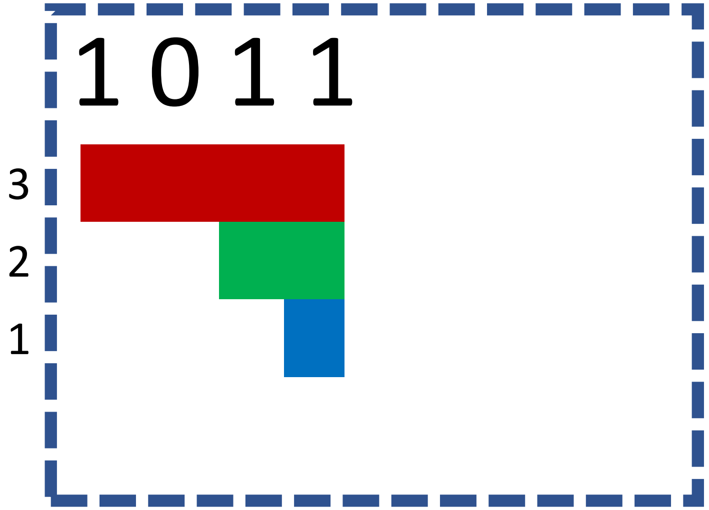
- Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- Each time there are three instances that report “close” values, delete the middle one
- Use a number of different starting points to “sandwich” the sliding window



- Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

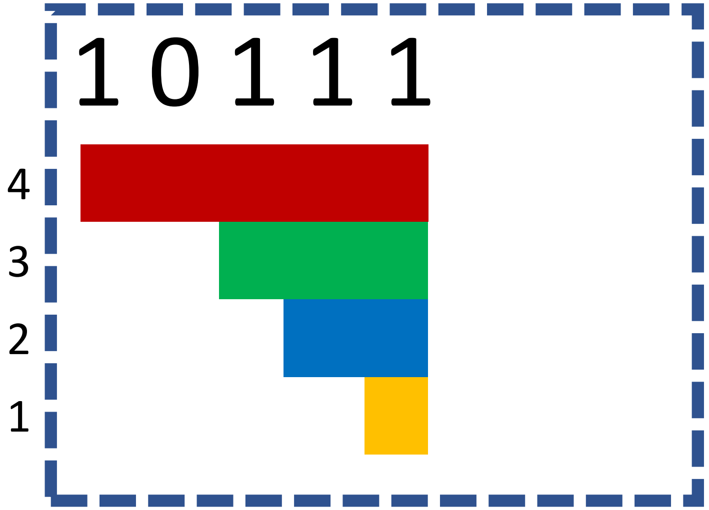
- ❖ Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- ❖ Each time there are three instances that report “close” values, delete the middle one
- ❖ Use a number of different starting points to “sandwich” the sliding window



- ❖ Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

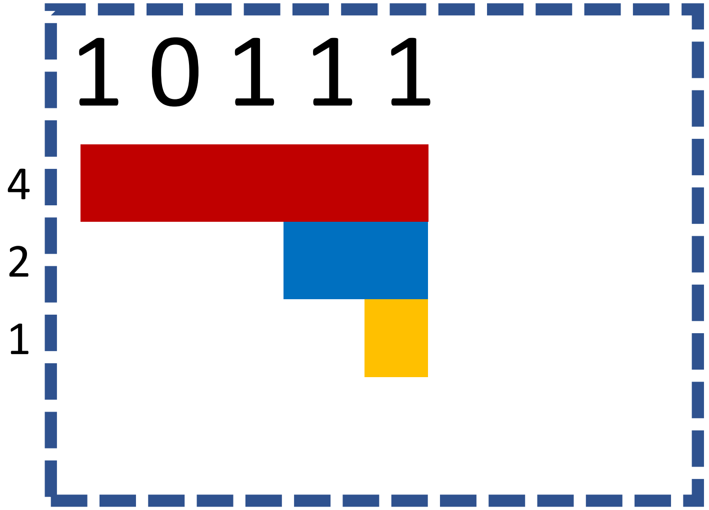
- ❖ Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- ❖ Each time there are three instances that report "close" values, delete the middle one
- ❖ Use a number of different starting points to "sandwich" the sliding window



- ❖ Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

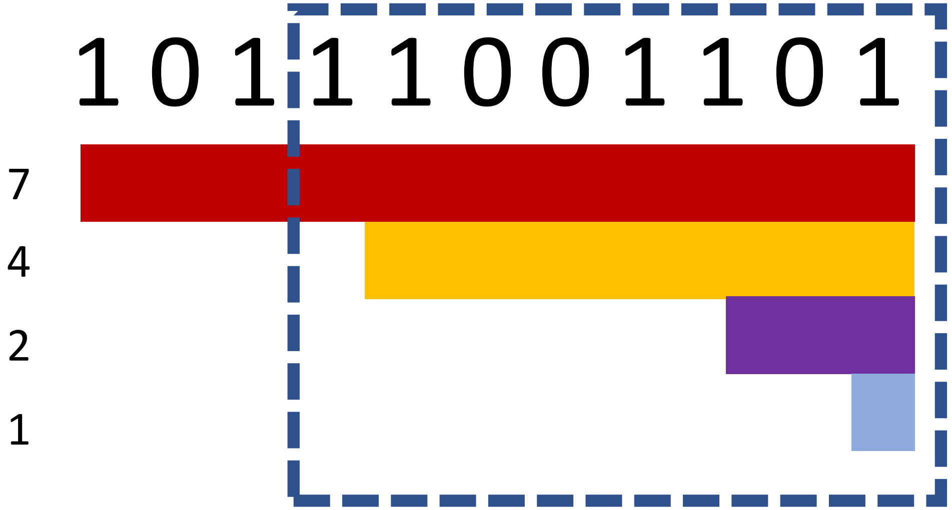
- ❖ Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- ❖ Each time there are three instances that report “close” values, delete the middle one
- ❖ Use a number of different starting points to “sandwich” the sliding window



- ❖ Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

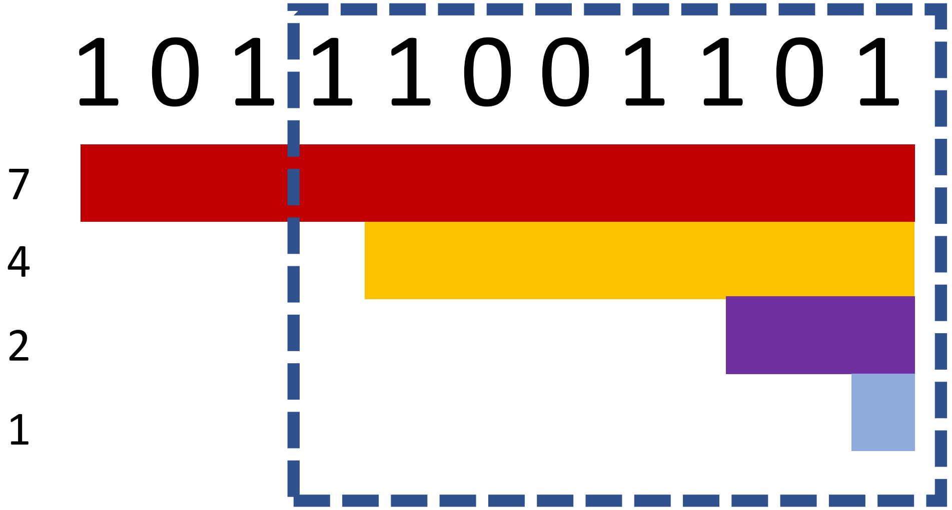
- ❖ Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- ❖ Each time there are three instances that report “close” values, delete the middle one
- ❖ Use a number of different starting points to “sandwich” the sliding window



- ❖ Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

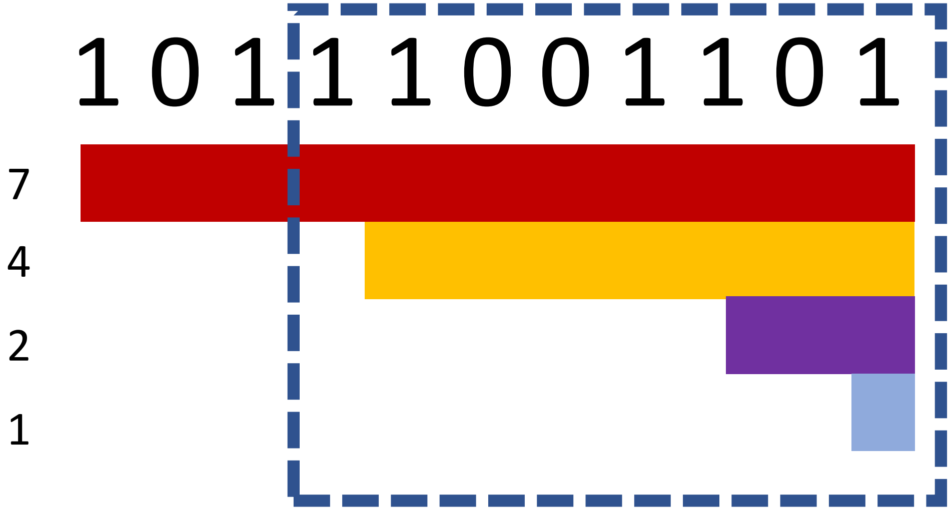
- Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- Each time there are three instances that report “close” values, delete the middle one
- Use a number of different starting points to “sandwich” the sliding window



- Example: Number of ones in sliding window (2-approximation)

Smooth Histogram

- Start a new instance of the streaming algorithm (along with existing instances) each time a new element arrives
- Each time there are three instances that report “close” values, delete the middle one
- Use a number of different starting points to “sandwich” the sliding window



- Example: Number of ones in sliding window (2-approximation)
- Number of ones in sliding window is at least 4 and at most 7
- 7 is a good approximation

Smooth Functions

- Quantiles, heavy-hitters, norm estimation, distinct elements, sampling
- Matchings, number of triangles, spanners, sparsifiers
- Numerical linear algebra (matrix multiplication, spectral approximation,…)
- Minimum enclosing ball, Clustering (k-means, k-median, k-centers,…)
- Submodular optimization
- Strings (pattern matching, periodicity, edit distance, Parikh tching,…)
- Codeword testing

Smooth Functions

- Quantiles, heavy-hitters, norm estimation, distinct elements, sampling
- Matchings, number of triangles, spanners, sparsifiers
- Numerical linear algebra (matrix multiplication, spectral approximation,…)
- Minimum enclosing ball, Clustering (k-means, k-median, k-centers,…)
- Submodular optimization
- Strings (pattern matching, periodicity, edit distance, Parikh matching,…)
- Codeword testing

Smooth Functions

- Quantiles, heavy-hitters, norm estimation, distinct elements, sampling
- Matchings, number of triangles, spanners, sparsifiers
- Numerical linear algebra (matrix multiplication, spectral approximation,…)
- Minimum enclosing ball, Clustering (k-means, k-median, k-centers,…)
- Submodular optimization
- Strings (pattern matching, periodicity, edit distance, Parikh matching,…)
- Codeword testing

Smooth Functions

[BGO13, BGLWZ18] [BOZ09]

- ❖ Quantiles, heavy-hitters, norm estimation, distinct elements, sampling
- ❖ Matchings, number of triangles, spanners, sparsifiers
- ❖ Numerical linear algebra (matrix multiplication, spectral approximation,…)
- ❖ Minimum enclosing ball, Clustering (k-means, k-median, k-centers,…) [BLLM16]
- ❖ Submodular optimization [CNZ16,ELVZ17]
- ❖ Strings (pattern matching, periodicity, edit distance, Parikh matching,…)
- ❖ Codeword testing

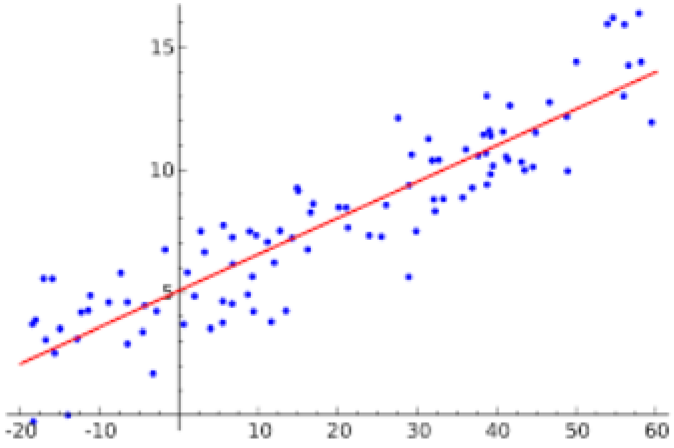
Smooth Functions

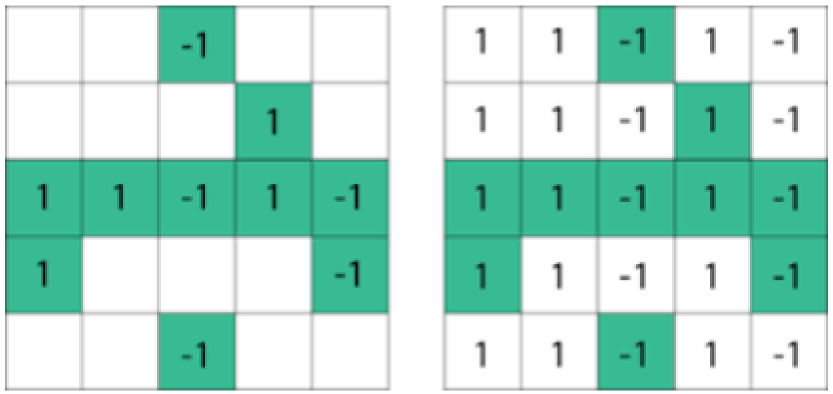
[BGO13, BGLWZ18] [BOZ09]

- Quantiles, heavy-hitters, norm estimation, distinct elements, sampling
- Matchings, number of triangles, spanners, sparsifiers
- Numerical linear algebra (matrix multiplication, spectral approximation,…)
- Minimum enclosing ball, Clustering (k-means, k-median, k-centers,…) [BLLM16]
- Submodular optimization [CNZ16,ELVZ17]
- Strings (pattern matching, periodicity, edit distance, Parikh matching,…)
- Codeword testing

Why Randomized Numerical Linear Algebra?

- ❖ Given massive sources of data
- ❖ Predication and Optimization: Principle Component Analysis (PCA), Low-Rank Approximation (LRA), Regression





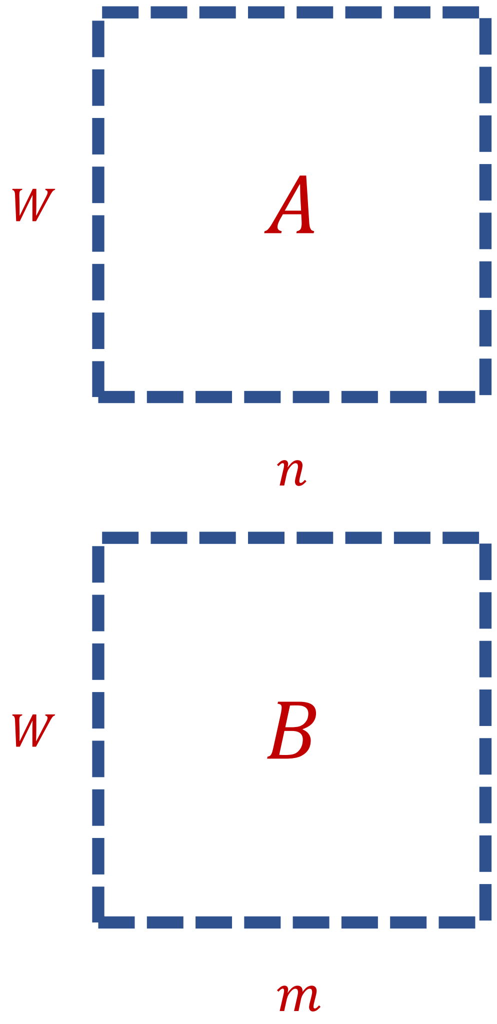
Linear Algebra Background

- Vectors $u, v \in R^n$
- Inner product: $\langle u, v \rangle = \sum u_i v_i \in R$
- Outer product: $u \otimes v = uv^T \in R^{n \times n}$

$$\begin{matrix} u_1 v_1 & u_1 v_2 & \cdots & u_1 v_n \\ u_2 v_1 & u_2 v_2 & \cdots & u_2 v_n \\ \vdots & \vdots & \ddots & \vdots \\ u_n v_1 & u_n v_2 & \cdots & u_n v_n \end{matrix}$$

Linear Algebra Background

- Matrices: $A \in R^{W \times n}, B \in R^{W \times m}$
- $(A^{\top}B)_{i,j} = \langle a_i b_j \rangle$, where a_i is the i^{th} column of A and b_j is the j^{th} column of B.
- $A^T B = \sum_{i=1}^{W} a_i \otimes b_i = \sum_{i=1}^{W} a_i^{\top} b_i$



Approximate Matrix Multiplication

- Vector norm: $\|x\|_p = \left(x_1^p + x_2^p + \cdots x_n^p\right)^{\frac{1}{p}}$
- Frobenius norm: $\|A\|_F = \sqrt{\sum A_{i,j}^2}$
- Matrices: $A \in R^{W \times n}, B \in R^{W \times m}, W \gg m, n$
- Output $A^T B$

- Can we find $C \in R^{d \times n}, D \in R^{d \times m}, d \ll W$ such that $C^\top D \approx A^\top B$?
- What does $\approx$ mean?

Approximate Matrix Multiplication

- Given $\epsilon > 0$, find $C \in R^{d \times n}, D \in R^{d \times m}$ such that

$$\|A^\top B - C^\top D\|_F \leq \epsilon \|A^\top B\|_F$$

- Information Retrieval: $A \in R^{W \times n}$ rows represent documents, columns represent occurrence of each word
- High entries of $AA^\top$ correspond to "similar" documents

Approximate Matrix Multiplication (Offline) [DK01]

- Given $\epsilon > 0$ and $A \in R^{W \times n}, W \gg n$, find $B \in R^{d \times n}$ such that

$$\|A^\top A - B^\top B\|_F \leq \epsilon \|A^\top A\|_F$$

- Intuition: Large entries in $A^\top A$ come from large entries in A
- Importance sampling: Sample row a_i of A proportional to its squared row norm
- Sample row a_i of A with probability $p_{\mathrm{i}} \propto \frac{\|a_i\|_2^2}{\|A\|_F^2}$
- Rescale each sampled row by $\frac{1}{\sqrt{p_i}}$

Approximate Matrix Multiplication (Offline)

- Analyze $\mathrm{E}\left[\|A^\top A - B^\top B\|^2{}_F\right]$

- Step 1: Show that $B^\top B$ is an unbiased estimator:
- $\mathrm{E}[B^\top B] = \sum p_k \left(\frac{1}{\sqrt{p_k}} a_k^\top \frac{1}{\sqrt{p_k}} a_k\right) = A^\top A$
- Step 2: Bound the variance of $(B^\top B)_{i,j}$:
- $\mathrm{Var}\left[(B^\top B)_{i,j}\right] \leq \sum \frac{1}{p_k} (a_k^\top a_k)_{i,j}^2$

Approximate Matrix Multiplication (Offline)

- $\mathrm{E}\left[\|A^\top A - B^\top B\|^2{}_F\right] = \mathrm{E}\left[\sum_{i,j}(A^\top A - B^\top B)_{i,j}^2\right]$
- $\mathrm{E}\left[\|A^\top A - B^\top B\|^2{}_F\right] = \sum_{i,j} \mathrm{Var}\left[(A^\top A - B^\top B)_{i,j}\right]$
- $\mathrm{E}\left[\|A^\top A - B^\top B\|^2{}_F\right] \le \sum_{i,j,k} \frac{1}{p_k}\left(a_k^\top a_k\right)_{i,j}^2 = \sum_k \frac{1}{p_k}\|a_k\|_2^4$

- For $p_k = \frac{c\|a_k\|_2^2}{\|A\|_F^2}$, $\mathrm{E}\left[\|A^\top A - B^\top B\|^2{}_F\right] \le \frac{1}{c}\|A\|_F^4$.
- $\sum p_\mathrm{k} = c := \frac{1}{\epsilon^2}$, so total number of sampled rows is $O\left(\frac{1}{\epsilon^2}\log n\right)$ whp

Randomized Numerical Linear Algebra (randNLA) on Sliding Windows

1	3	5	-2	7	0	11	4	-8
0	0	-1	3	13	2	8	6	2
2	5	6	1	4	0	-7	5	3
8	7	2	1	-1	-3	-2	-4	-6
-5	3	-4	-1	-2	-1	0	-3	-1
7	1	3	2	4	1	0	11	1

- Various stream update models
- In this talk, rows arrive one-by-one in the data stream

Randomized Numerical Linear Algebra (randNLA) on Sliding Windows

1	3	5	-2	7	0	11	4	-8

0	0	-1	3	13	2	8	6	2
2	5	6	1	4	0	-7	5	3
8	7	2	1	-1	-3	-2	-4	-6
-5	3	-4	-1	-2	-1	0	-3	-1
7	1	3	2	4	1	0	11	1
1	2	5	-5	4	1	23	4	-3

- Rows arrive one-by-one in the data stream

Randomized Numerical Linear Algebra (randNLA) on Sliding Windows

1	3	5	-2	7	0	11	4	-8
0	0	-1	3	13	2	8	6	2
2	5	6	1	4	0	-7	5	3
8	7	2	1	-1	-3	-2	-4	-6
-5	3	-4	-1	-2	-1	0	-3	-1
7	1	3	2	4	1	0	11	1
1	2	5	-5	4	1	23	4	-3
0	5	0	0	7	0	1	31	6

- Rows arrive one-by-one in the data stream

Approximate Matrix Multiplication

- Goal: Sample row a_i of A with probability $p_\mathrm{i} \propto \frac{\|a_i\|_2^2}{\|A\|_F^2}$

- See a_i in the sliding window model, can compute $\|a_i\|_2^2$
- Cannot compute $\|A\|_F^2$ without seeing all the rows

- How would we do matrix multiplication in the streaming model?

Approximate Matrix Multiplication

- How would we do matrix multiplication in the streaming model?
- Track $\|A\|_F^2$
- Suppose we have sampled row a_i of A with probability $p_{\mathrm{i}} \propto \frac{\|a_i\|_2^2}{\|A\|_F^2}$
- New row arrives a_t: $\|A\|_F^2$ increases by $\|a_t\|_2^2$
- What do we do with a_i?
- Downsample: keep a_i with probability $\frac{\|A\|_F^2}{\|A\|_F^2+\|a_t\|_2^2}$
- Sampled a_i with probability $p_{\mathrm{i}} \propto \frac{\|a_i\|_2^2}{\|A\|_F^2+\|a_t\|_2^2}$

Approximate Matrix Multiplication

- Note it suffices to have $\widehat{A}$ a 2-approximation of $\|A\|_F^2$
- Why? Sample row a_i of A with probability $p_{\mathrm{i}} \propto \frac{2\|a_i\|_2^2}{\widehat{A}}$
- Frobenius norm is *smooth*
- Use smooth histogram to maintain $\widehat{A}$

- Smooth histogram for Frobenius norm
- Separate instance of matrix multiplication streaming algorithm for each instance tracking the Frobenius norm

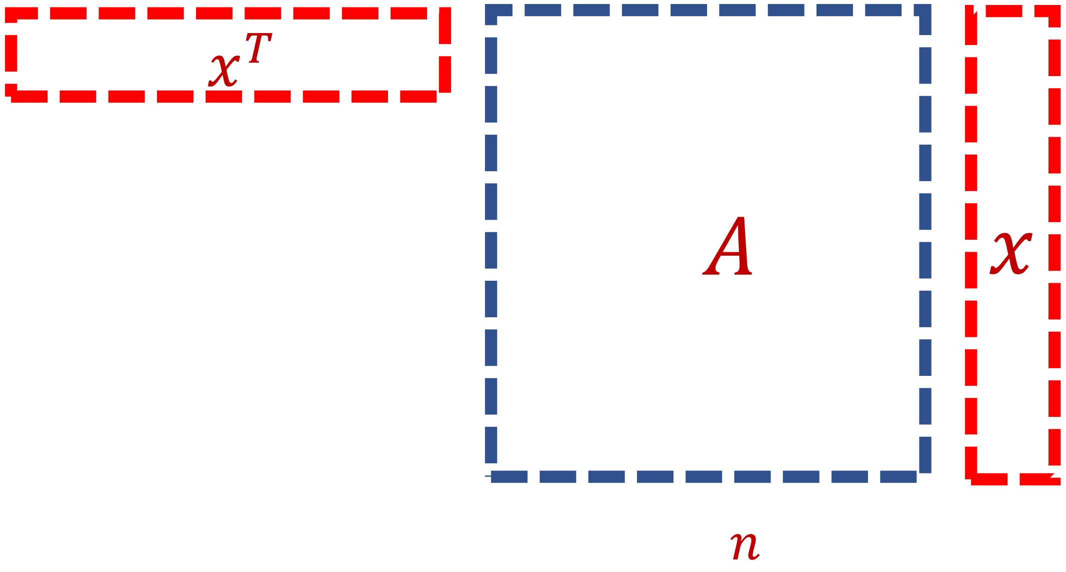
Approximate Matrix Multiplication

- Total space: $O\left(\frac{1}{\epsilon^2}\log n\right)$ rows $\rightarrow$ $O\left(\frac{n}{\epsilon^2}\log^2 n\right)$ bits of space
- Can decrease to $O\left(\frac{n}{\epsilon^2}\log n\left(\log\log n+\log\frac{1}{\epsilon}\right)\right)$ with bit representation tricks

- Also give $\Omega\left(\frac{n}{\epsilon^2}\log n\right)$ space lower bound

Questions?

Spectral Sparsification



- Find a matrix B so that for all vectors x, $x^T Bx$ is a good approximation for $x^T Ax$
- Approximates *all* cuts of a graph

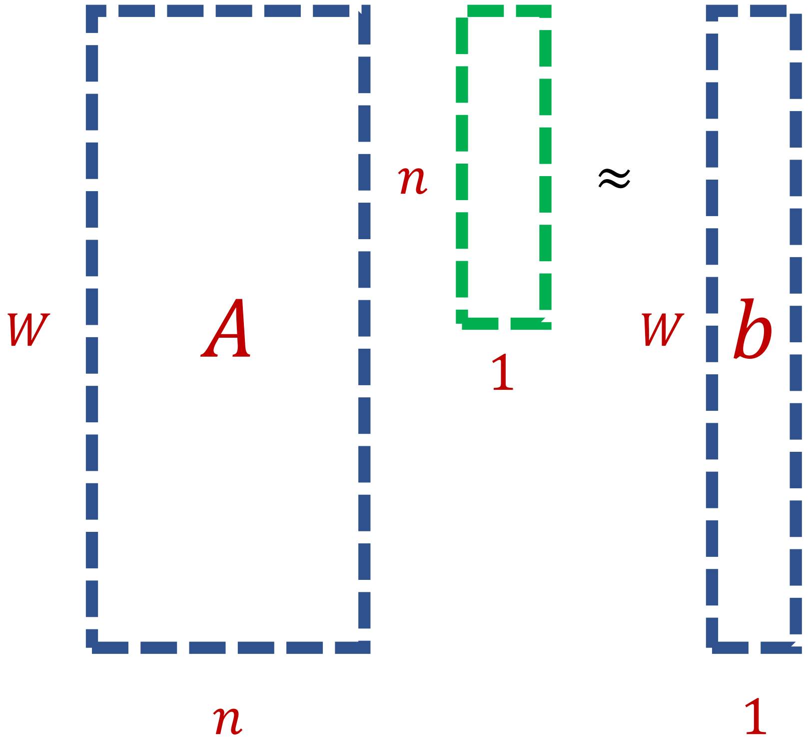
Spectral Approximation

- Spectral approximation: Given $\epsilon > 0$ and $A \in R^{W \times n}$, find matrix $M \in R^{m \times n}$ with $m \ll W$, such that for *every* $x \in R^n$,

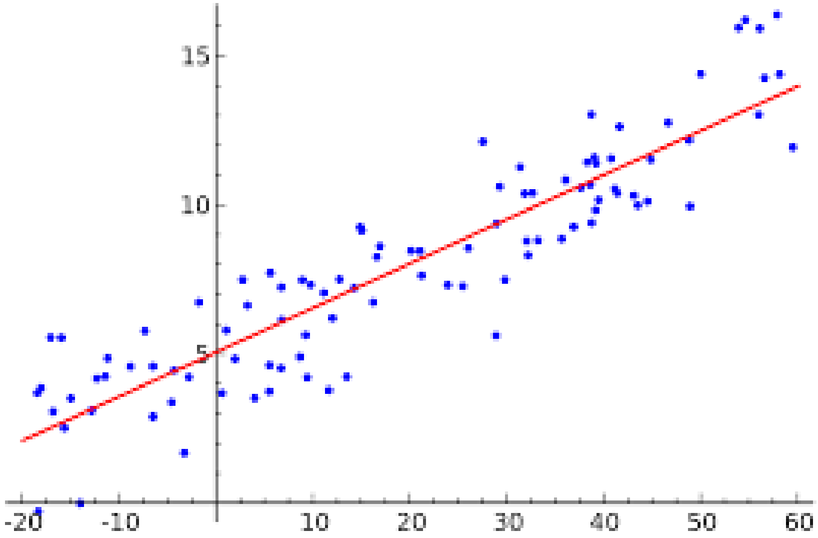
$$(1-\epsilon)\|Ax\|_2 \le \|Mx\|_2 \le (1+\epsilon)\|Ax\|_2$$

- Eigenvalue: $Ax = \lambda x$
- Singular value: square root of eigenvalue of $A^T A$
  - $\sigma_1(A) \ge \sigma_2(A) \ge \cdots \ge \sigma_n(A)$
- Spectral approximation gives approximation of all the eigenvalues
- Schatten p norm: $\|A\|_p = \left(\sigma_1^p + \sigma_2^p + \cdots \sigma_n^p\right)^{\frac{1}{p}}$

Regression



- Find the vector x that minimizes $\|Ax - b\|_2$
- "Least squares" optimization



Linear Algebra Background

- A symmetric matrix $M \in R^{n \times n}$ is positive semi-definite (PSD) if $x^\top M x \geq 0$ for all vectors $x \in R^n$
- All eigenvalues of PSD matrix M are non-negative

- If $A - B$ is PSD, we write $B \preccurlyeq A$

- For any vector $v \in R^n$, $v^\top v$ is a PSD matrix
- Sum of two PSD matrices is a PSD matrix

Linear Algebra Background

- Spectral approximation: Given $\epsilon > 0$ and $A \in R^{W \times n}$, find matrix $M \in R^{m \times n}$ with $m \ll W$, such that for *every* $x \in R^n$,

$$(1-\epsilon)\|Ax\|_2 \leq \|Mx\|_2 \leq (1+\epsilon)\|Ax\|_2$$

- Equivalent to $(1-\epsilon)A^\top A \preccurlyeq M^\top M \preccurlyeq (1+\epsilon)A^\top A$

Initial Approach (Smooth PSD Histogram)

- Recall smooth function: If $f(A)$ is a "good" approximation to $f(B)$, then $f(A \cup C)$ will always be a "good" approximation to $f(B \cup C)$.
- Monotonic, polynomially bounded, all properties of real values…

- Use partial (Loewner) ordering on PSD matrices
  - If matrix $A \in R^{r \times n}$ is a submatrix of $B \in R^{s \times n}$, then $A^\top A \preccurlyeq B^\top B$
- The singular values of $A^\top A$ are respectively at most those of $B^\top B$

- Have "monotonicity", what about smoothness?

Initial Approach (Smooth PSD Histogram)

- If $(1-\epsilon)B^\top B \preccurlyeq A^\top A \preccurlyeq (1+\epsilon)B^\top B$, then for any matrix C,

$$(1-\epsilon)(B^\top B + C^\top C) \preccurlyeq A^\top A + C^\top C \preccurlyeq (1+\epsilon)(B^\top B + C^\top C)$$

- The singular values of the matrices behave “smoothly”
- Maintain histogram based on the singular values

- Each substream represents a matrix A
- Keep $A^\top A$ and merge whenever there are three matrices within $(1+\epsilon)$ in Loewner ordering

Initial Approach (Smooth PSD Histogram)

- Space? Each instance stores a matrix $A^{\top}A$
- A has at most W rows but $A^{\top}A \in R^{n\times n}$
- How many instances? n singular values, each of them polynomially bounded
- $O\left(\frac{1}{\epsilon} n \log n\right)$ instances

- Total space: $O\left(\frac{1}{\epsilon} n^3 \log n\right)$ (in words)

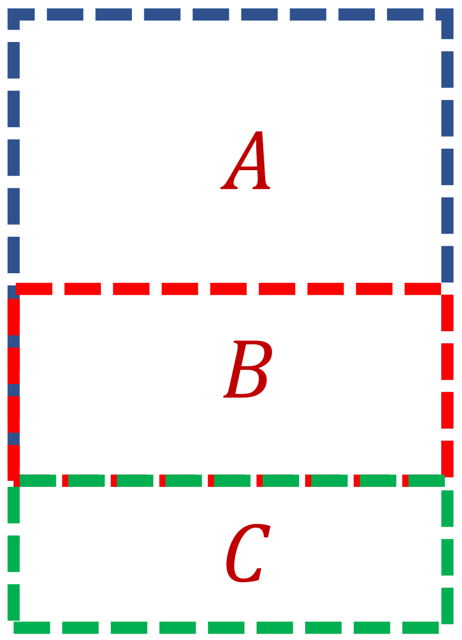
Initial Approach (Smooth PSD Histogram)

- Deterministic algorithm: $O\left(\frac{1}{\epsilon}n^3 \log n\right)$ space (in words)

- Outputs spectral approximation of $A^\top A$ rather than A
- Does not preserve sparsity
- Can be done in $\tilde{O}\left(\frac{1}{\epsilon^2}n^2\right)$ space in streaming

Intuition

- To decrease the space, we first observe there is a lot of similar structure between instances A, B, C: most rows are shared!
- Try subsampling approach?
- Squared row norm doesn't work…



Spectral Approximation (Offline)

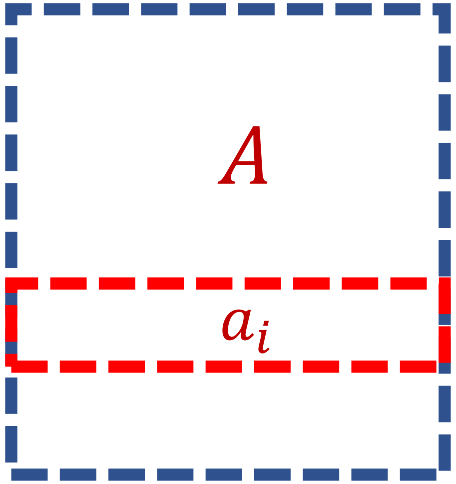
- Spectral approximation: Given $\epsilon > 0$ and $A \in R^{W \times n}$, find matrix $M \in R^{m \times n}$ with $m \ll W$, such that for *every* $x \in R^n$,

$$(1-\epsilon)\|Ax\|_2 \leq \|Mx\|_2 \leq (1+\epsilon)\|Ax\|_2$$

- How would we do this offline? Hint: fundamental tool of dimensionality reduction
- Although Johnson-Lindenstrauss reduces the number of rows and sparse JL can preserve sparsity, we want to focus on sampling

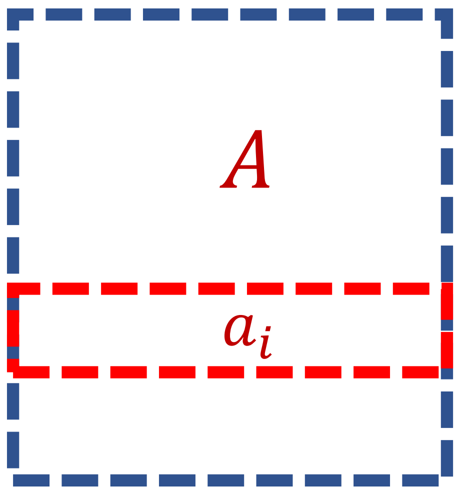
Linear Algebra Background

- Singular Value Decomposition (SVD): $A = U\Sigma V^T \in R^{W\times n}$
  - $U \in R^{W\times W}$ is an orthonormal matrix (rows, columns orthonormal)
  - $\Sigma \in R^{W\times n}$ is a rectangular diagonal matrix with non-negative entries
  - $V \in R^{n\times n}$ is an orthonormal matrix (rows, columns orthonormal)
- $\|u_i\|_2^2$ are the *leverage scores* of A (in this case of row a_i)
- Intuition: how "unique" a row is (recall importance sampling)
- $\ell_i = \max \frac{\langle a_i, x\rangle^2}{\|Ax\|_2^2} = \|u_i\|_2^2 = a_i(A^\top A)^{-1}a_i^\top$
- $\sum \ell_i = n$



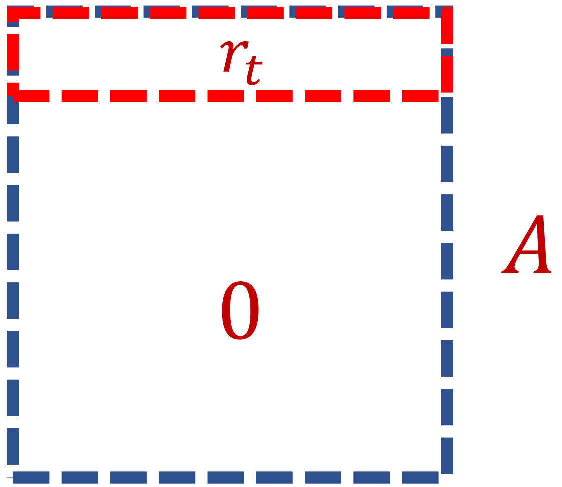
Spectral Approximation (Offline)

- Sample each row a_i with probability $p_i \propto \ell_i = a_i (A^\top A)^{-1} a_i^\top$
- Outputs a matrix $M \in R^{m \times n}$ such that $(1-\epsilon)\|Ax\|_2 \leq \|Mx\|_2 \leq (1+\epsilon)\|Ax\|_2$ for all $x \in R^n$ [DMM06, SS08]
- $\sum \ell_i = n$, so the total number of rows sampled is $\propto \tilde{O}(n)$
- Leverage scores are monotonic with more rows, so the offline approach can be adapted to streaming through downsampling



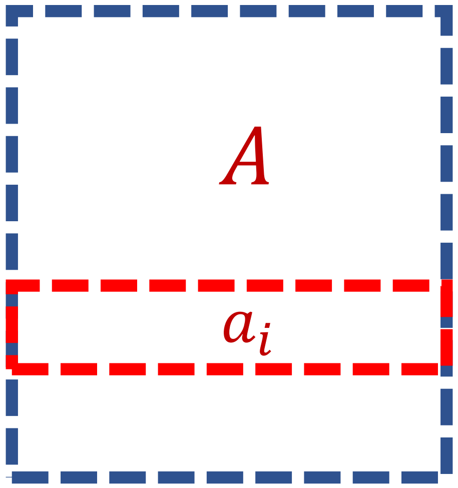
Spectral Approximation (Sliding Window)

- Consider the sliding window model: we see rows $r_1, r_2, \ldots$
- Leverage score of r_t tells us r_t is not important, so we do not sample r_t
- Stream proceeds: All rows before r_t expire, new rows are all zeros
- Cannot possibly get approximation of A without storing r_t
- This implies we should *always* store the most recent row!
- Need a new sense of importance accounting for both uniqueness AND recency of a row

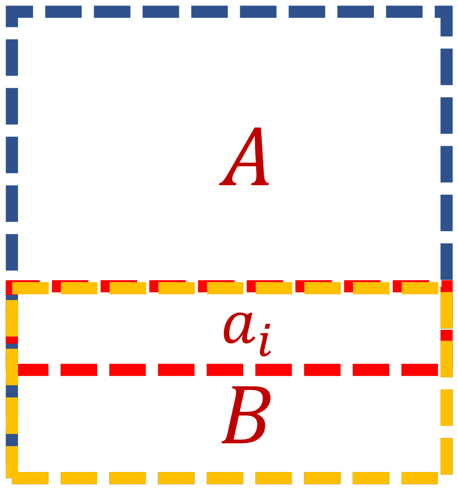


Reverse Online Leverage Scores

- Leverage score of row a_i is $\ell_i = a_i(A^\top A)^{-1}a_i^\top$
- Rows before a_i might be deleted so they shouldn't count towards the importance of a_i

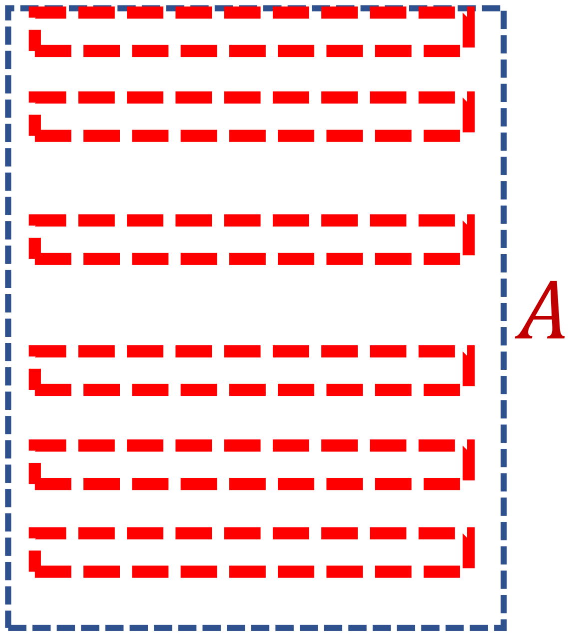


- Reverse online leverage score of row a_i is $\tau_i = a_i(B^\top B)^{-1}a_i^\top$ where B are the rows after a_i



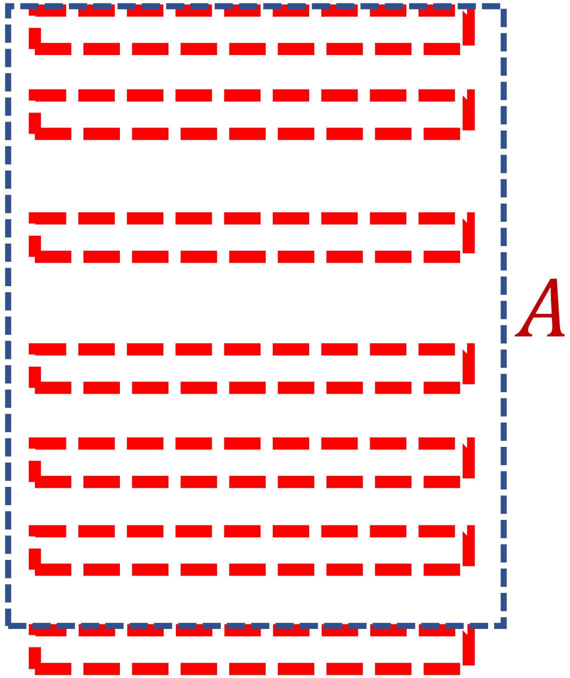
Algorithm

- Algorithm: sample (and rescale) a number of rows



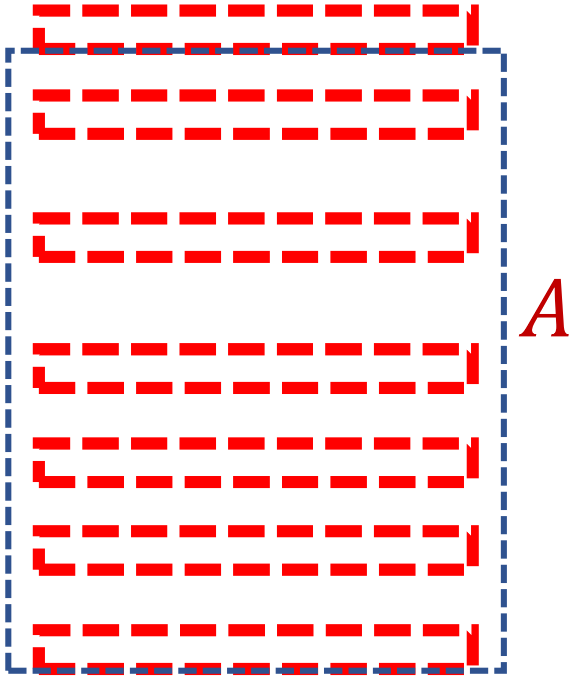
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it



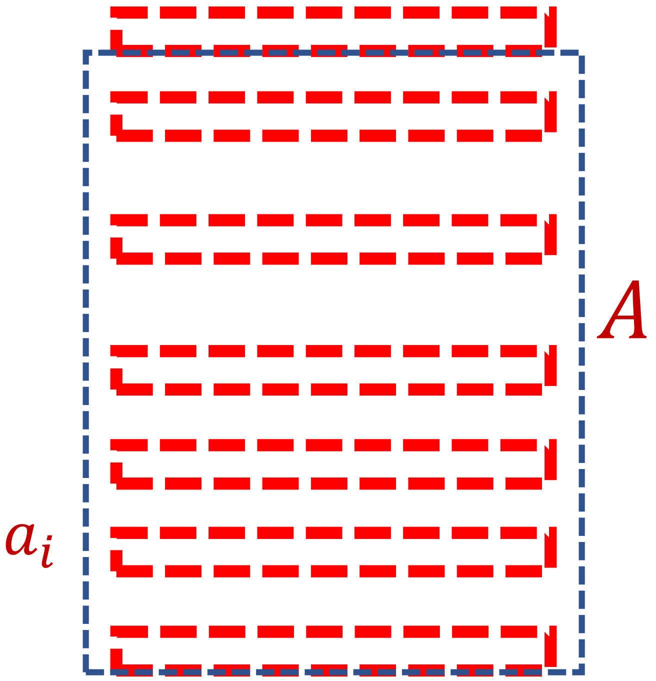
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it



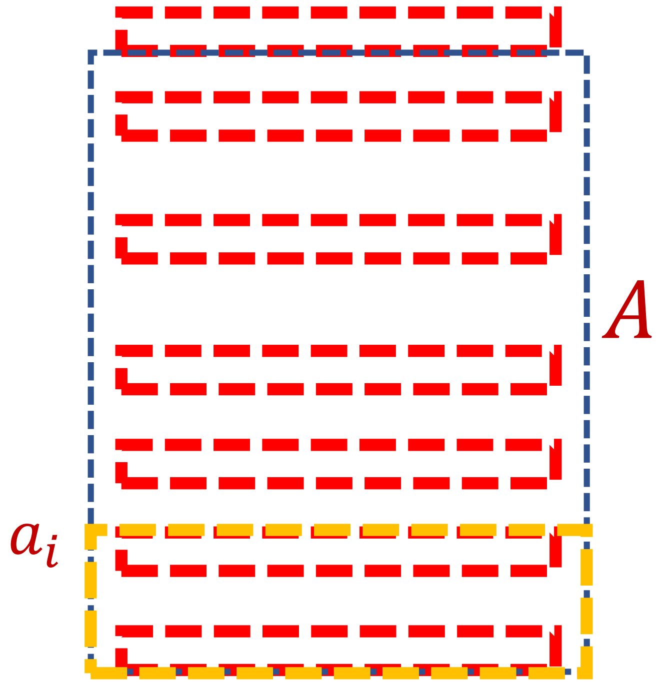
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



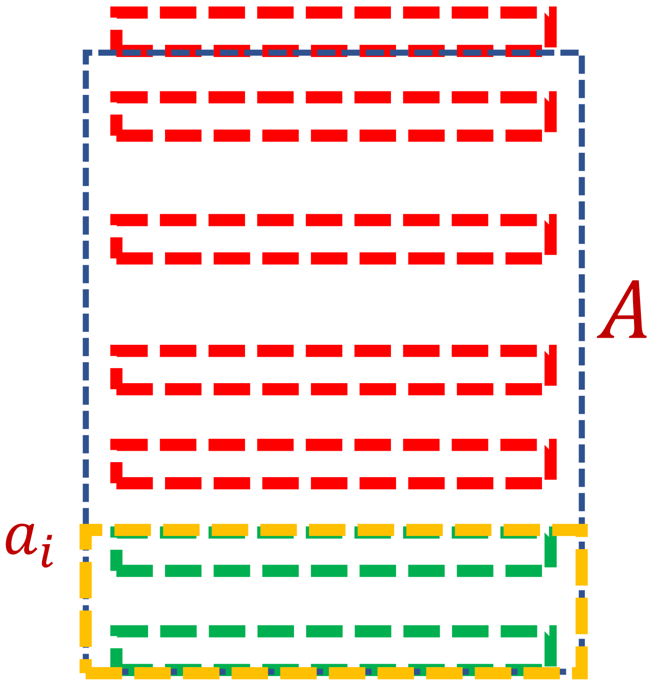
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



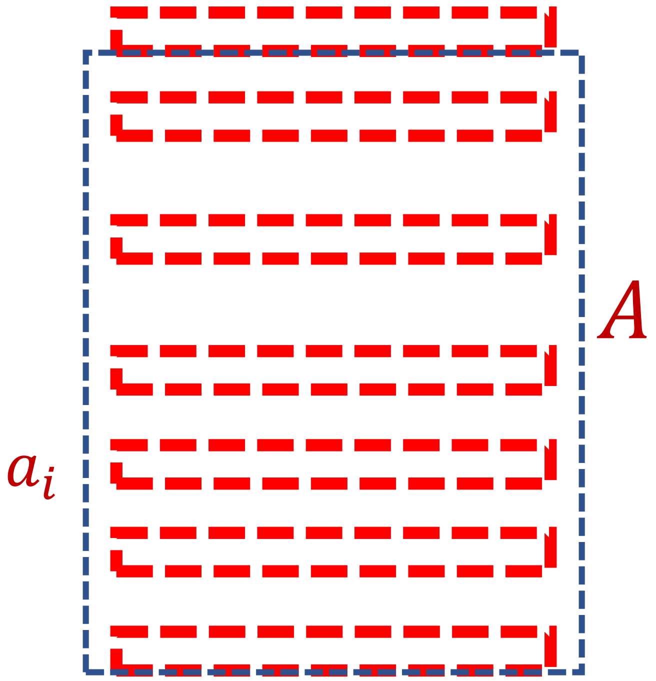
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



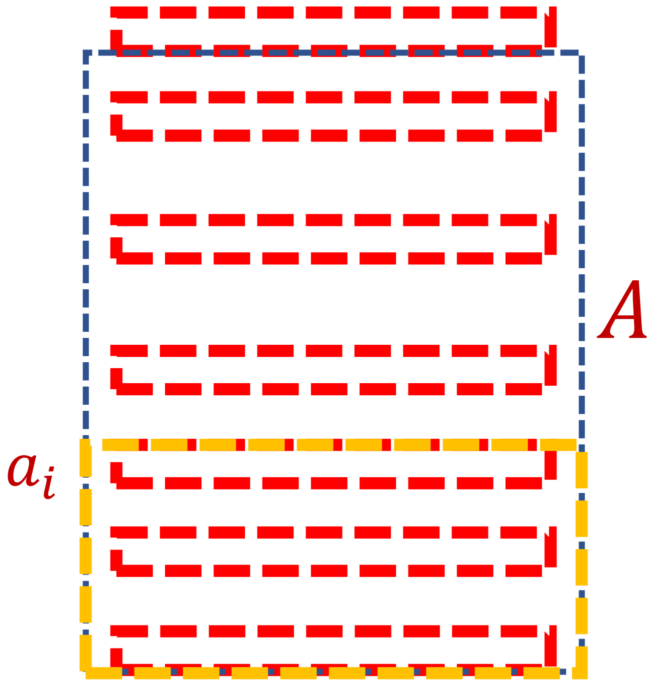
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



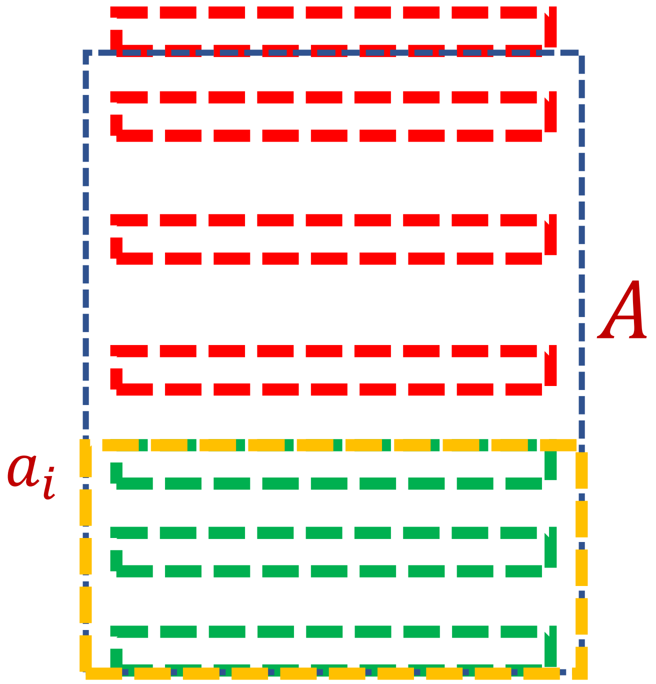
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



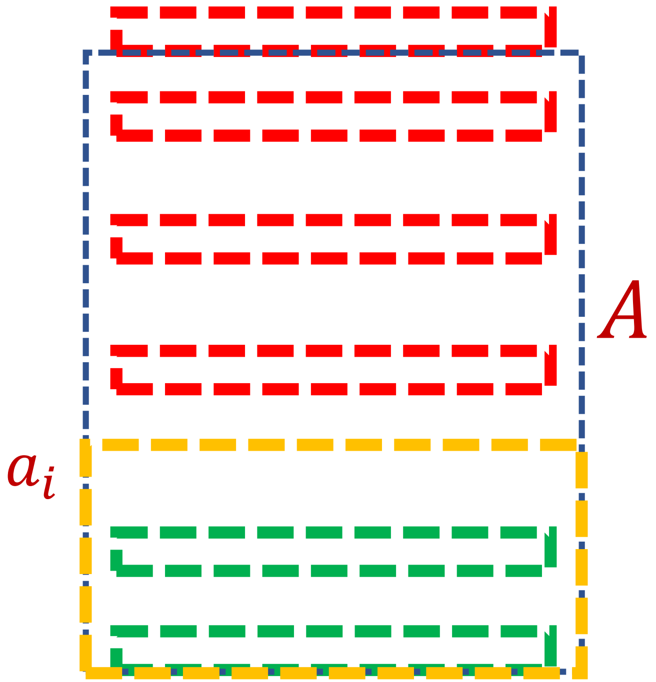
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



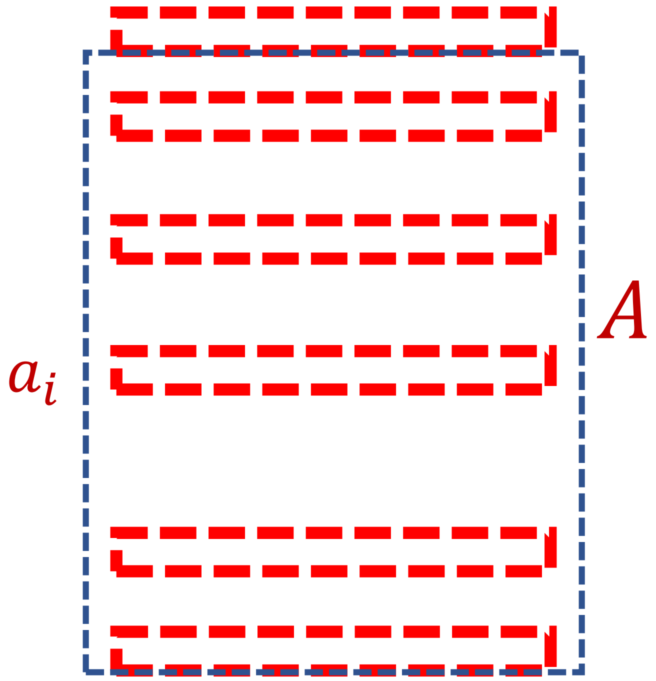
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



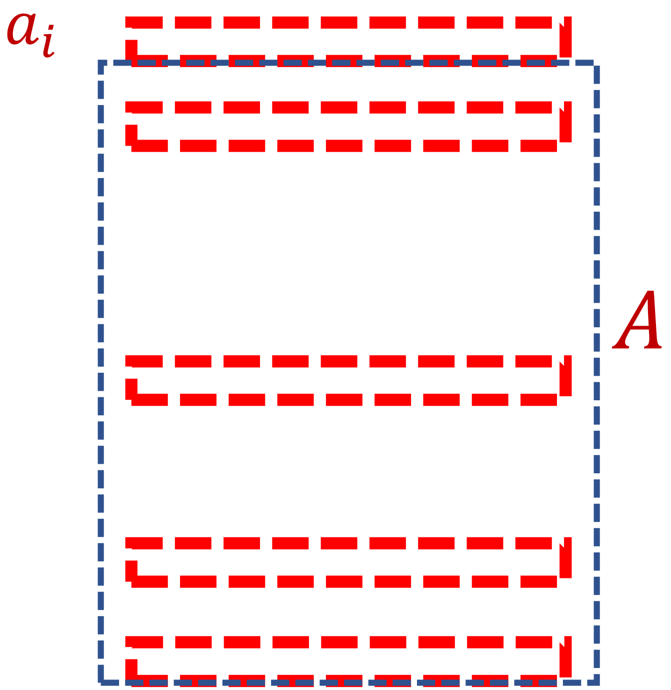
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



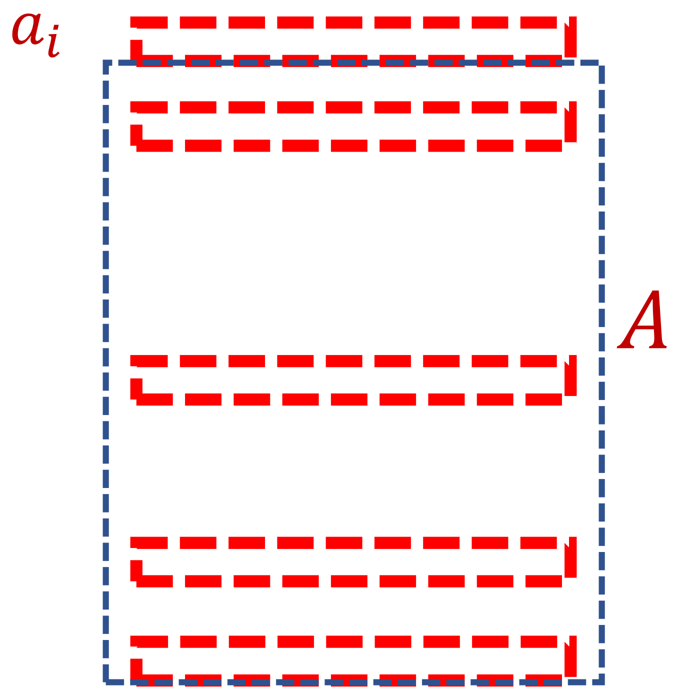
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling



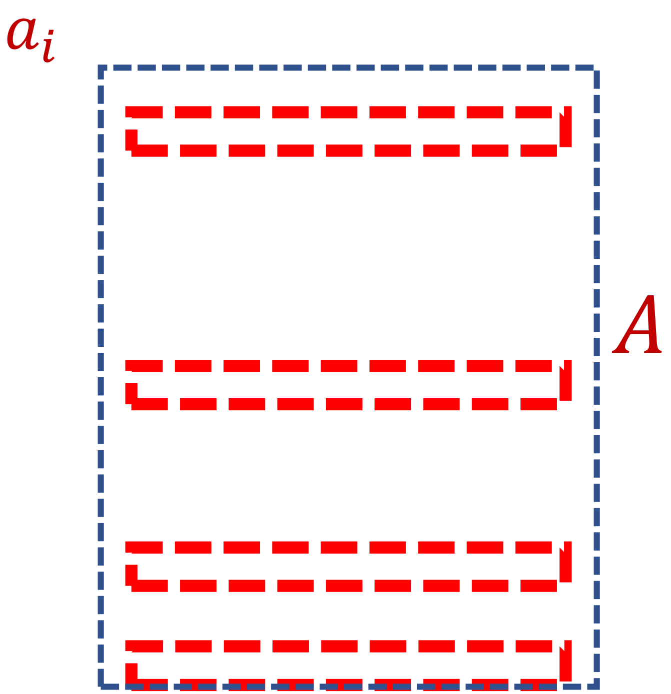
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling
- Delete expired rows



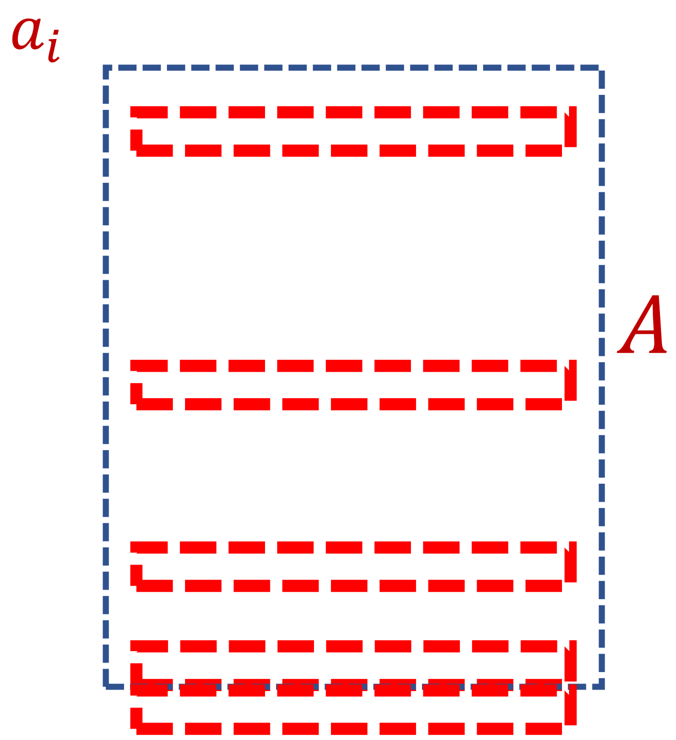
Algorithm

- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling
- Delete expired rows



Algorithm

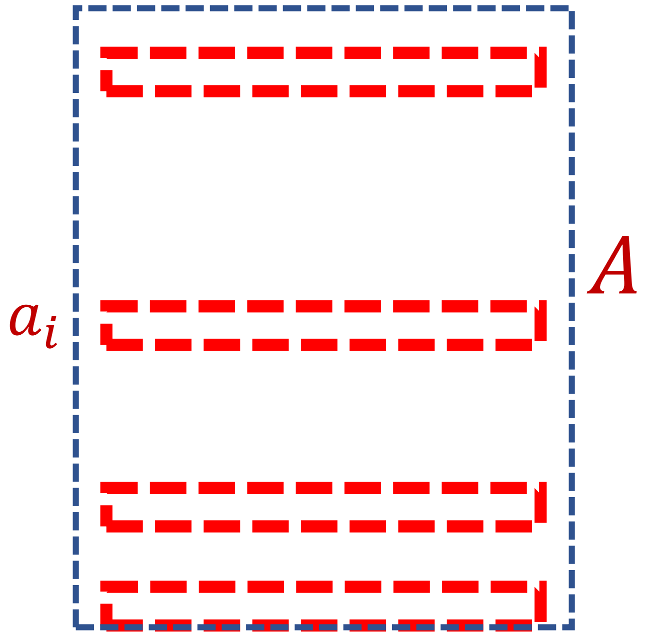
- Algorithm: sample (and rescale) a number of rows
- New row arrives – store it
- For each sampled (and rescaled) row a_i, sample the row with probability $\propto \tau_i \leftrightarrow$ downsampling
- Delete expired rows
- New row arrives – repeat



Algorithm

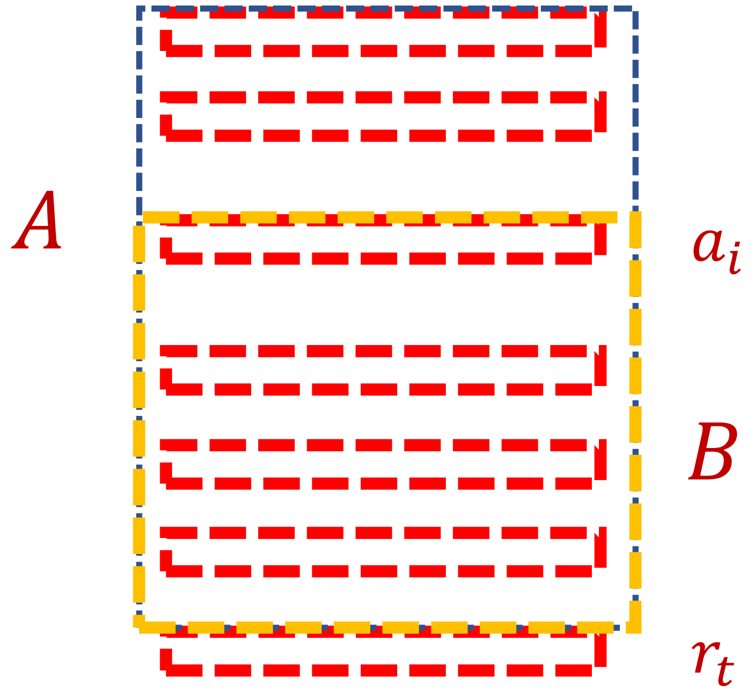
- Correctness: Show an invariant that each row a_i is sampled with probability $\propto$ *final* reverse online leverage score

- Outputs a matrix $M \in R^{m \times n}$ such that $(1-\epsilon)\|Ax\|_2 \leq \|Mx\|_2 \leq (1+\epsilon)\|Ax\|_2$ for all $x \in R^n$ [DMM06, SS08]



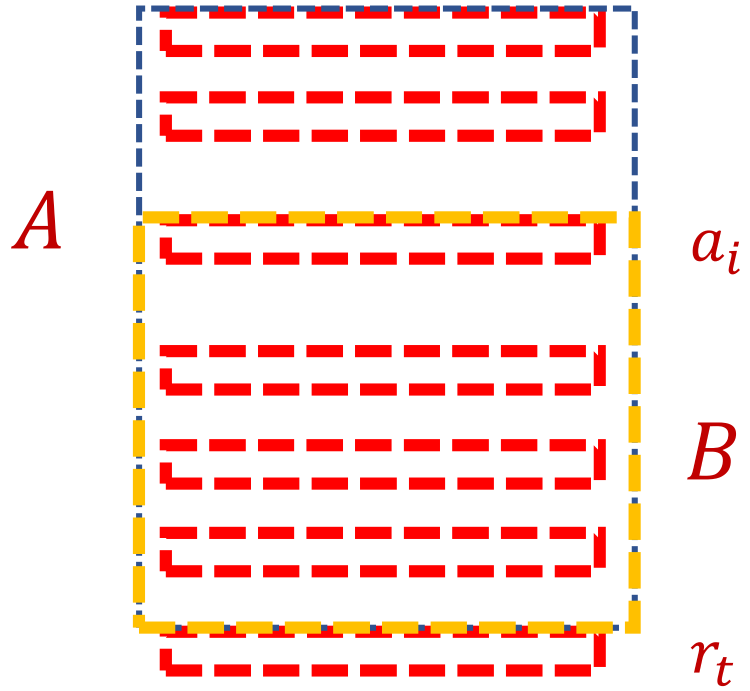
Correctness

- Correctness: Show an invariant that each row a_i is sampled with probability $\propto$ *final* reverse online leverage score
- Let B be the rows after a_i before row r_t



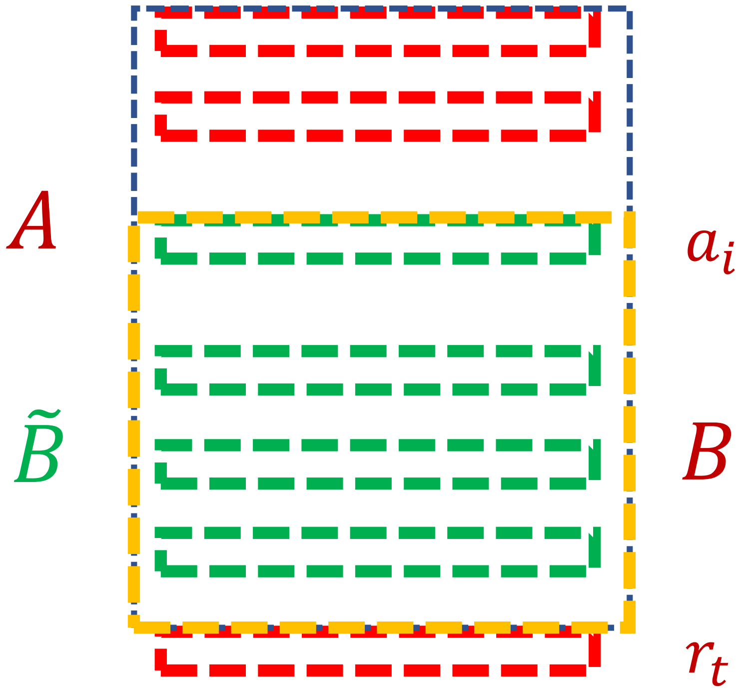
Correctness

- Correctness: Show an invariant that each row a_i is sampled with probability $\propto$ *final* reverse online leverage score
- Let B be the rows after a_i before row r_t
- Suppose before the arrival of row r_t, row a_i has been sampled with probability p_i, where $c_1\tau_B(a_i) \le p_i \le c_2\tau_B(a_i)$



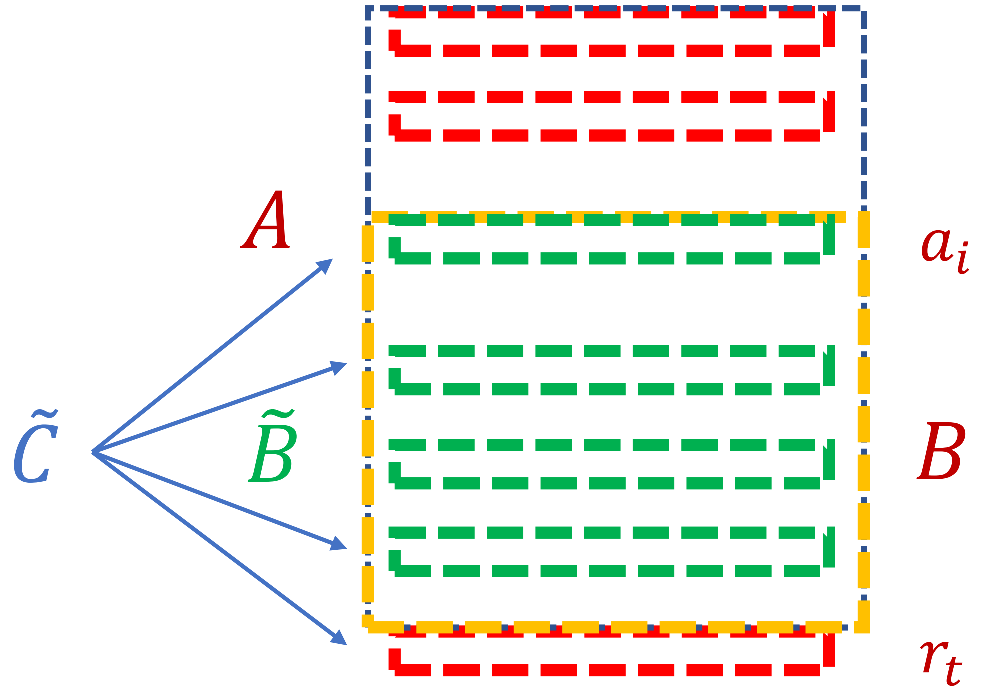
Correctness

- Suppose before the arrival of row r_t, row a_i has been sampled with probability p_i, where $c_1\tau_B(a_i) \leq p_i \leq c_2\tau_B(a_i)$
- $(1-\epsilon)B^\top B \preccurlyeq \tilde{B}^\top\tilde{B} \preccurlyeq (1+\epsilon)B^\top B$ [DMM06, SS08]



Correctness

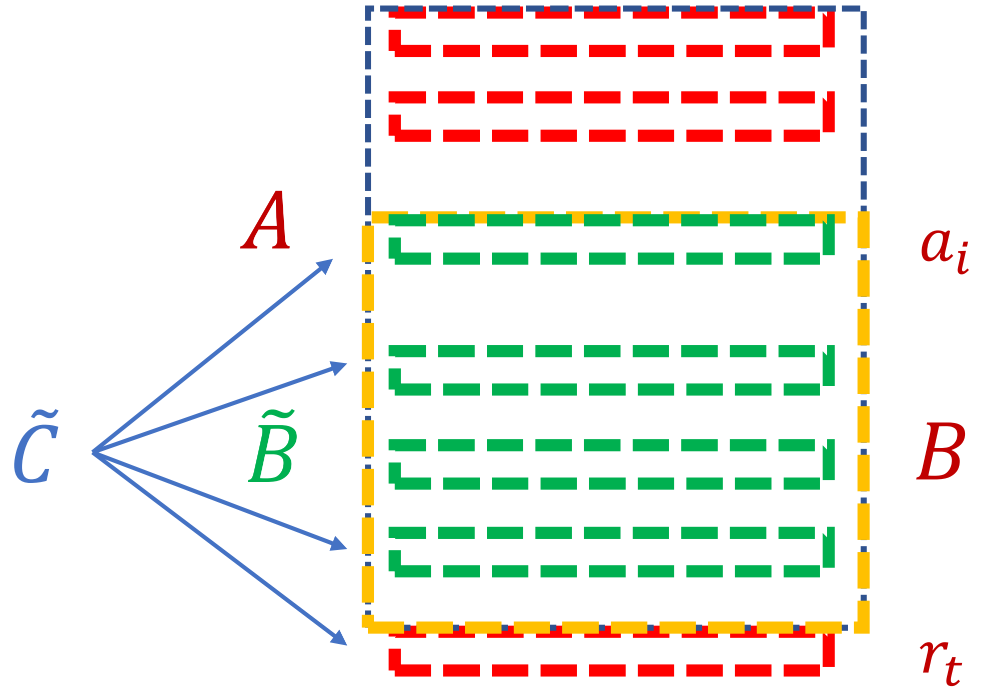
- Suppose before the arrival of row r_t, row a_i has been sampled with probability p_i, where $c_1\tau_B(a_i) \leq p_i \leq c_2\tau_B(a_i)$
- $(1-\epsilon)B^\top B \preccurlyeq \tilde{B}^\top \tilde{B} \preccurlyeq (1+\epsilon)B^\top B$ [DMM06, SS08]
- Let $\tilde{C}$ be $\tilde{B}$ appended by r_t
- a_i remains with probability $\propto \tau_{\tilde{C}}\left(\frac{a_i}{\sqrt{p_i}}\right)$



Correctness

- a_i remains with probability $\propto \tau_{\tilde{C}}\left(\frac{a_i}{\sqrt{p_i}}\right)$
- Reverse online leverage score:

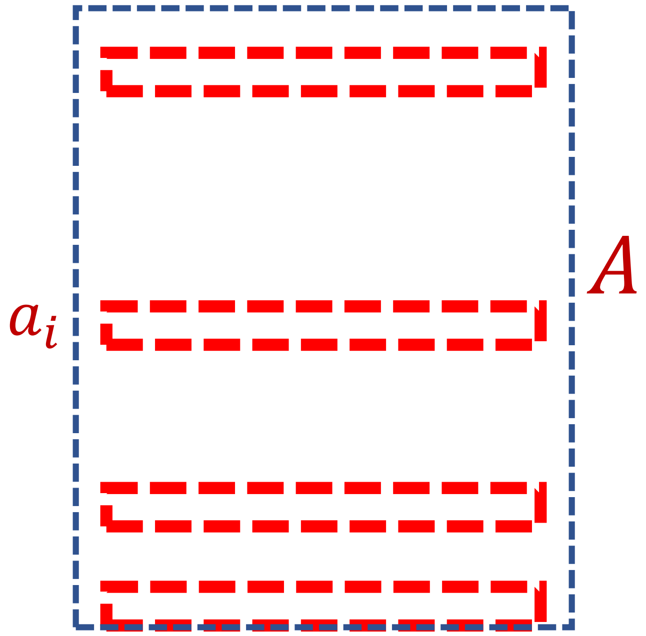
$$\left(\frac{a_i}{\sqrt{p_i}}\right)(\tilde{C}^\top\tilde{C})^{-1}\left(\frac{a_i}{\sqrt{p_i}}\right)^\top = \left(\frac{a_i}{\sqrt{p_i}}\right)(\tilde{B}^\top\tilde{B} + r_t^\top r_t)^{-1}\left(\frac{a_i}{\sqrt{p_i}}\right)^\top$$



- Recall $(1-\epsilon)B^\top B \preccurlyeq \tilde{B}^\top\tilde{B} \preccurlyeq (1+\epsilon)B^\top B$, so $(1-\epsilon)C^\top C \preccurlyeq \tilde{C}^\top\tilde{C} \preccurlyeq (1+\epsilon)C^\top C$
- a_i survives w.p. $c_1\tau_C(a_i) \le p_i \le c_2\tau_C(a_i)$

Algorithm

- Correctness: Show an invariant that each row a_i is sampled with probability $\propto$ *final* reverse online leverage score
- By monotonicity, a_i is sampled with probability $\propto$ leverage score
- Outputs a matrix $M \in R^{m \times n}$ such that $(1-\epsilon)\|Ax\|_2 \leq \|Mx\|_2 \leq (1+\epsilon)\|Ax\|_2$ for all $x \in R^n$ [DMM06, SS08]



- Space? Must bound $\sum \tau_i$

Reverse Online Leverage Scores

- Online algorithm: see rows sequentially and irrevocably store or discard row, output spectral approximation at the end
- Sum of reverse online leverage scores = sum of online leverage scores [CMP16]
- $\sum \tau_i = \tilde{O}\left(\frac{1}{\epsilon^2} n\right) \rightarrow \tilde{O}\left(\frac{1}{\epsilon^2} n^2\right)$ space algorithm

Questions?

Low-Rank Approximation

$$
\begin{array}{cc}
 & \begin{array}{ccccccccc}
1 & 3 & 5 & -2 & 7 & 0 & 11 & 4 & -8 \\
0 & 0 & -1 & 3 & 13 & 2 & 8 & 6 & 2 \\
2 & 5 & 6 & 1 & 4 & 0 & -7 & 5 & 3 \\
8 & 7 & 2 & 1 & -1 & -3 & -2 & -4 & -6 \\
-5 & 3 & -4 & -1 & -2 & -1 & 0 & -3 & -1 \\
7 & 1 & 3 & 2 & 4 & 1 & 0 & 11 & 1
\end{array} \\
w & \\
 & n
\end{array}
$$

- Find rank k matrix A_k that minimizes $\|A_k - A\|_F$
- Finding structure among noise
- Matrix completion problem

NETFLIX

Low-Rank Approximation

- Low-rank approximation: Given $\epsilon > 0$ and $A \in R^{W \times n}$, find matrix $M \in R^{m \times n}$ with $m \ll W$ such that

$$(1-\epsilon)\|A - A_k\|_F \leq \|M - M_k\|_F \leq (1+\epsilon)\|A - A_k\|_F$$

Low-Rank Approximation (Offline)

- SVD: $A = U\Sigma V^T$ with singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$
- Let Σ_k be the matrix with diagonal entries $\sigma_1, \ldots, \sigma_k$
- $M = U\Sigma_k V^T$ is *optimal* solution

Low-Rank Approximation

- Spectral algorithm solves low-rank approximation: $\tilde{O}\left(\frac{1}{\epsilon^2}n^2\right)$ space

- Can be done in $\tilde{O}\left(\frac{1}{\epsilon^2}kn\right)$ space in streaming

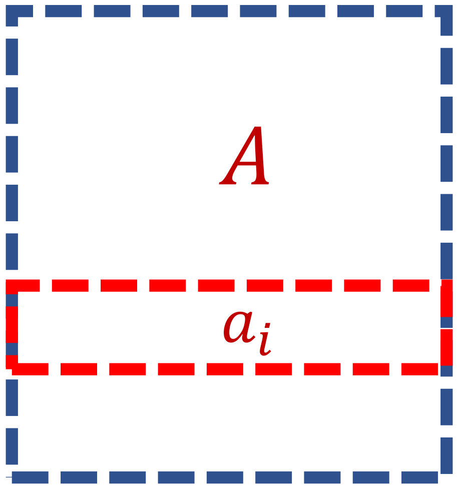
Low-Rank Approximation (Streaming)

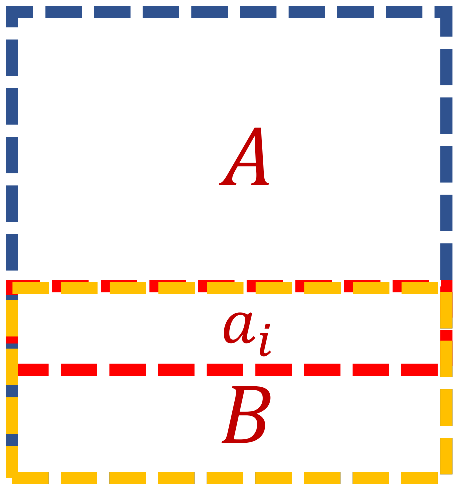
- Leverage score: $a_i(A^\top A)^{-1}a_i^\top$
- Ridge leverage score: $\ell_i = a_i(A^\top A + \lambda I_n)^{-1}a_i^\top$, where $\lambda = \frac{\|A-A_k\|_F^2}{k}$

- Sample each row a_i with probability $p_i \propto \ell_i$
- Outputs a matrix $M \in R^{m\times n}$ such that $(1-\epsilon)\|A - A_k\|_F \leq \|M - M_k\|_F \leq (1+\epsilon)\|A - A_k\|_F$ [CMM15]
- $\sum \ell_i = 2k$ [CMM15]

Template

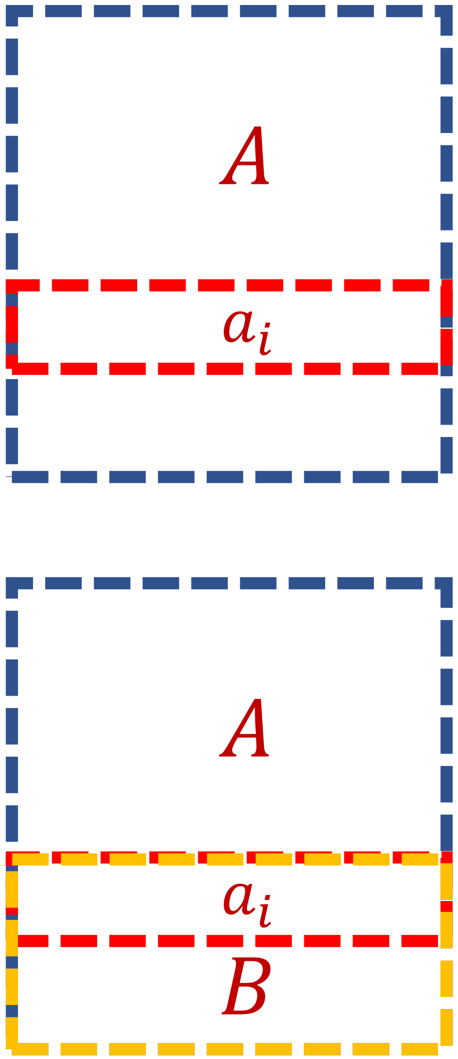
- Suppose we know $\lambda = \frac{\|A-A_k\|_F^2}{k}$
- Reverse online leverage score: Sample each row a_i with probability $p_i \propto \tau_i = a_i(B^\top B + \lambda I_n)^{-1}a_i^\top$
- Monotonicity of ridge leverage score
- Outputs a matrix $M \in R^{m\times n}$ such that $(1-\epsilon)\|A-A_k\|_F \leq \|M-M_k\|_F \leq (1+\epsilon)\|A-A_k\|_F$ [CMM15]





Template

- Issue #1: Compute $\lambda = \frac{\|A - A_k\|_F^2}{k}$
- Issue #2: Bound $\sum \tau_i$

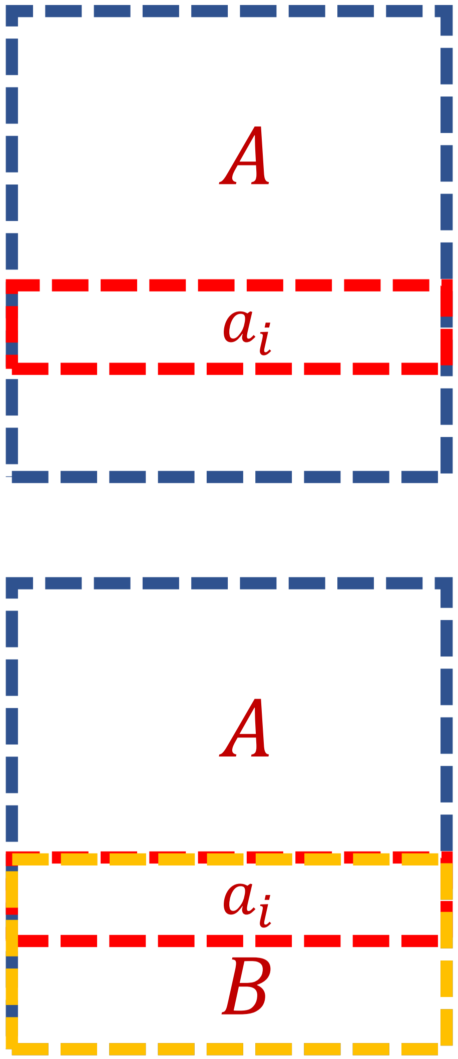


Regularization Computation

- Observation: it suffices to have a constant factor approximation of $\lambda = \frac{\|A - A_k\|_F^2}{k}$
- Use projection-cost preserving sketch [CEMMP15] to reduce the dimension of each row
- Feed reduced rows into spectral approximation algorithm

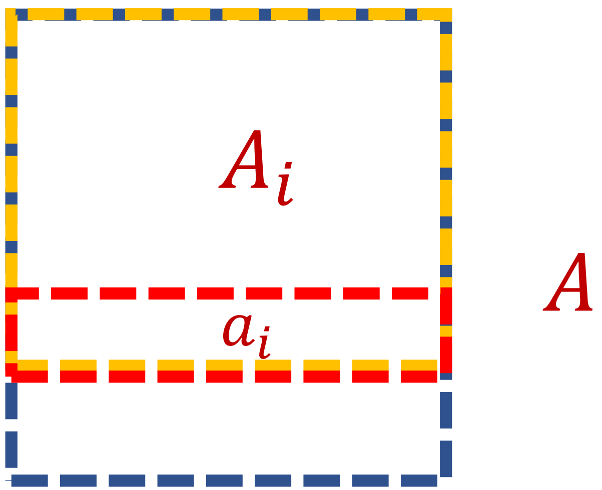
Template

- ✓ Issue #1: Compute $\lambda = \frac{\|A-A_k\|_F^2}{k}$
- ❖ Issue #2: Bound $\sum \tau_i$



Reverse Online Leverage Scores

- Let A_i be the first i rows of A
- Bound sum of $\tau_i = a_i\left(A_i^\top A_i + \lambda I_n\right)^{-1} a_i^\top$



Matrix Determinant Lemma

- $\det(A + v^\top v) = \det(A)(1 + vA^{-1}v^\top)$
- $\tau_i = a_i\left(A_i^\top A_i + \lambda I_n\right)^{-1} a_i^\top$

Using Matrix Determinant Lemma [CMP16]

- ❖ $\det(A + v^\top v) = \det(A)(1 + vA^{-1}v^\top)$
- ❖ $\tau_i = a_i\left(A_i^\top A_i + \lambda I_n\right)^{-1} a_i^\top$

$$\begin{aligned}\det(A^\top A + \lambda I_n) &= \det(A_{W-1}^\top A_{W-1} + \lambda I_n)\,(1 + a_W(A_{W-1}^\top A_{W-1} + \lambda I_n)^{-1} a_W^\top) \\ &= \det(A_{W-1}^\top A_{W-1} + \lambda I_n)\,(1 + \tau_{\mathrm{W}}) \\ &\geq \det(A_{W-1}^\top A_{W-1} + \lambda I_n)\left(1 + e^{\tau_W/2}\right)\end{aligned}$$

$$\det(A^\top A + \lambda I_n) \geq \lambda^n\, e^{\sum \tau_i/2}$$

Bounding the Determinant

- $\det(A^\top A + \lambda I_n) = \prod \sigma_i(A^\top A + \lambda I_n)$
- Small singular values: $\sigma_{k+1} + \ \ldots + \sigma_n = \|A - A_k\|_F^2 + \lambda(n-k)$
- By AM-GM,

$$\prod_{i=k+1}^{i=n} \sigma_i \leq \left(\frac{\|A - A_k\|_F^2 + \lambda(n-k)}{n-k}\right)^{n-k}$$

- Large singular values: $\sigma_i \leq \|A\|_2^2 + \lambda$ for $1 \leq i \leq k$

$$\log\det(A^\top A + \lambda I_n) = O(k \log n)$$

Reverse Online Leverage Scores

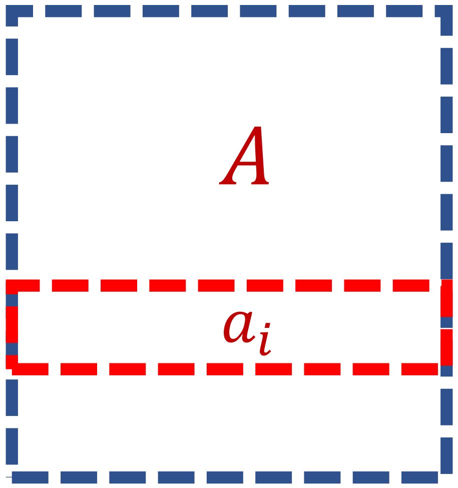
- $\det(A^\top A + \lambda I_n) \geq \lambda^n \, e^{\sum \tau_i / 2}$
- $\log \det(A^\top A + \lambda I_n) = O(k \log n)$

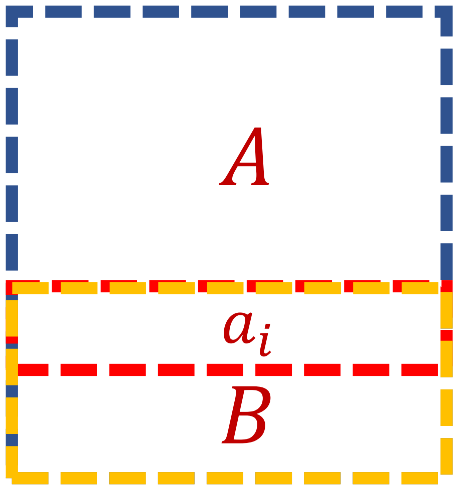
- $\sum \tau_i = O(k \log n)$

- Also gives a space efficient *online* algorithm for low-rank approximation!
- Can use slightly different estimator for $\lambda = \frac{\|A - A_k\|_F^2}{k}$ [AN13]

Template

- ✓ Issue #1: Compute $\lambda = \frac{\|A - A_k\|_F^2}{k}$
- ✓ Issue #2: Bound $\sum \tau_i$





Results

- Smooth histogram does not work for: vector induced p norms, generalized regression, low-rank approximation
- (Vector induced p norm: $||A||_p = \max||Ax||_p$ for $||x||_p = 1$)

Problem	Space	Reference
Deterministic Spectral Approximation	$\tilde{\mathcal{O}}\left(\frac{n^3}{\varepsilon}\right)$	Theorem 1.1
Spectral Approximation	$\tilde{\Theta}\left(\frac{n^2}{\varepsilon^2}\right)$	Theorem 4.5
Rank k Approximation	$\tilde{\Theta}\left(\frac{nk}{\varepsilon^2}\right)$	Theorem 5.8
Online Rank k Approximation	$\tilde{\Theta}\left(\frac{nk}{\varepsilon^2}\right)$	Theorem 6.4
Covariance Matrix Approximation	$\tilde{\Theta}\left(\frac{n}{\varepsilon^2}\right)$	Theorem 6.20, Theorem 6.24

Results

- If the entries of A and x are bounded integers, $O\left(\operatorname{poly}\left(n, \frac{1}{\epsilon}\right)\right)$ space algorithm for ℓ_1 spectral approximation:

$$(1-\epsilon)\|Ax\|_1 \leq \|Mx\|_1 \leq (1+\epsilon)\|Ax\|_1$$

- Algorithms can be slightly modified to run in *input sparsity time*
  - Only require constant factor approximation to reverse online leverage score
  - Use sparse JL for subspace embedding

Questions?

thank you
danke
謝謝
рахмат
ngiyabonga
teşekkür ederim
Баярлалаа
спасибо
faafetai lava
mersi
kia ora
barka
welalin
tack
spas
dank je
misaotra
matondo
paldies
grazzi
mahalo
tapadh leat
gracias
vinaka
спасибі
blagodaram
kiitos
dankie
nanni
nandri
dhanyavad
bedankt
enkosi
bayarlalaa
gràçie
hvala
mauruuru
köszönöm
akun
dankon
ačiū
хвала
asante
manana
obrigada
murakoze
tenki
djiere dieuf
tau
mochchakkeram
дякую
mamnun
chokrane
dziękuję
chnorakaloutioun
gratias ago
gràcies
sulpáy
taiku
go raibh maith agat
sobodi
děkuji
obrigado
sagolun
sukriya
kop khun krap
mèsi
didi madloba
najis tuke
ありがとう
arigatô
takk
dakujem
trugarez
kam sah hamnida
terima kasih
tanemirt
grazie
rahmet
dhanyavadagalu
shukriya
mercé
мерси
rahmat
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
diolch
merci

ℓ_1 Spectral Approximation

- Given $\epsilon > 0$ and $A \in R^{W \times n}$, find matrix $M \in R^{m \times n}$ with $m \ll W$, such that for *every* $x \in R^n$

$$(1-\epsilon)\|Ax\|_1 \le \|Mx\|_1 \le (1+\epsilon)\|Ax\|_1$$

- Robust to outliers, but unstable solution and possibly multiple solutions

ℓ_1 Leverage Scores

- Previous ℓ_2 leverage scores: $\ell_i = \max \frac{\langle a_i, x\rangle^2}{\|Ax\|_2^2}$
- ℓ_1 leverage scores: $\ell_i = \max \frac{|\langle a_i, x\rangle|}{\|Ax\|}$
- Sample each row a_i with probability $p_i \propto \ell_i$ gives ℓ_1 spectral approximation [DDHKM07]
- Bound the sum of the reverse online ℓ_1 leverage scores

ℓ_1 Leverage Scores

- Make nice assumptions: the entries of A and x are bounded integers
- Can show that if $||Ax||_1$ increases by $(1+\epsilon)$, $||Ax||_2^2$ must increase by $\left(1+\frac{\epsilon}{\mathrm{poly}(n)}\right)$
- Can use deterministic algorithm to find these breakpoints

- Use separate instances of streaming ℓ_1 spectral approximation algorithm starting at each of these breakpoints [DDHKM07, CP15]

Matrix Multiplication Lower Bounds

- Distributional INDEX: $\{0,1\}^n \times [n]$
- Alice has string $S \in \{0,1\}^n$ chosen uniformly at random
- Bob has index $i \in [n]$ chosen uniformly at random and must output $S[i]$ with probability $\frac{2}{3}$
- Requires $\Omega(n)$ bits of communication from Alice to Bob [MNSW98]

Matrix Multiplication Lower Bounds

- Alice has $S \in \{0,1\}^{n/c^2\epsilon^2}$
- Creates matrix $M \in \{-c, c\}^{\frac{1}{c^2\epsilon^2} \times n}$ in the natural way (stuffs string into matrix by sign)
- Creates matrix $A = [M \quad E]$, where E is $c^2\epsilon^2 n$ instances of e_1, …, $c^2\epsilon^2 n$ instances of e_{c^2/ϵ^2}
- If $\|A^\top A - B^\top B\|_F \leq \epsilon \|A^\top A\|_F$, then Bob can recover the signs of most entries in M and hence can recover most of the symbols of S

Future Work?

- Time decay models instead of sliding window
  - Polynomial decay, exponential decay,…
- Entrywise ℓ_1 low-rank approximation
- Other update models for sliding windows (ex. entrywise)
- Tighter bounds for Schatten-norm approximation