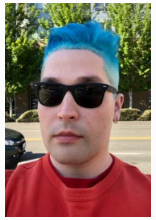
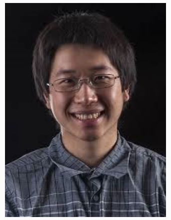
# Non-Adaptive Adaptive Sampling on Turnstile Streams


Sepideh Mahabadi
TTIC



Ilya Razenshteyn
MSR Redmond


David Woodruff
CMU



Samson Zhou
CMU

# Adaptive Sampling

An **algorithmic paradigm** for solving many data summarization tasks.

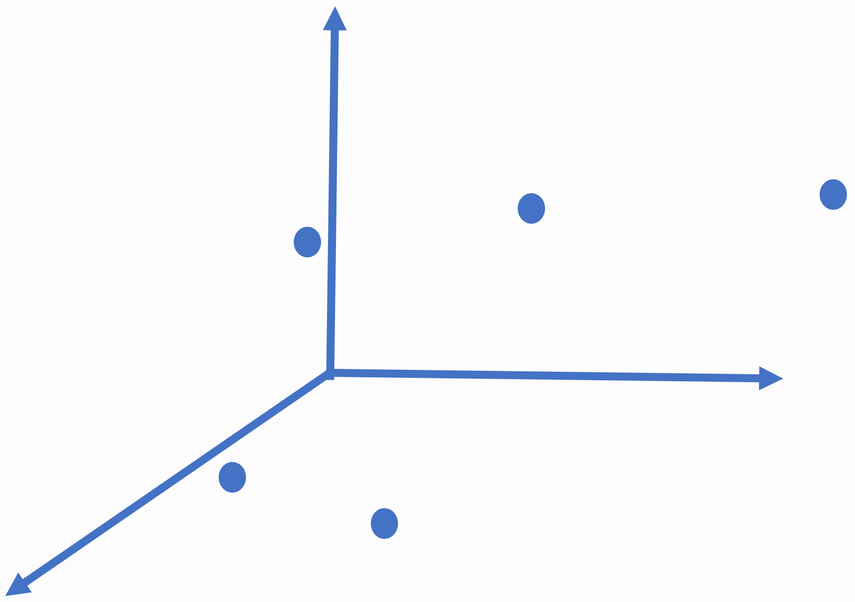
# Adaptive Sampling

An **algorithmic paradigm** for solving many data summarization tasks.

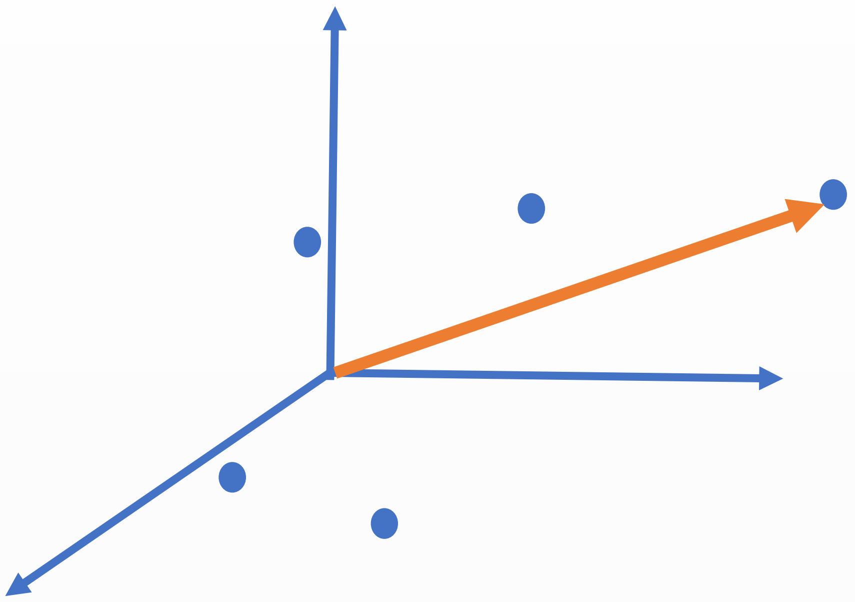
**Given:** $n$ vectors in $\mathbb{R}^d$

- **Sample** a vector w.p. proportional to its norm
- **Project** all vectors away from the selected subspace
- **Repeat** on the residuals

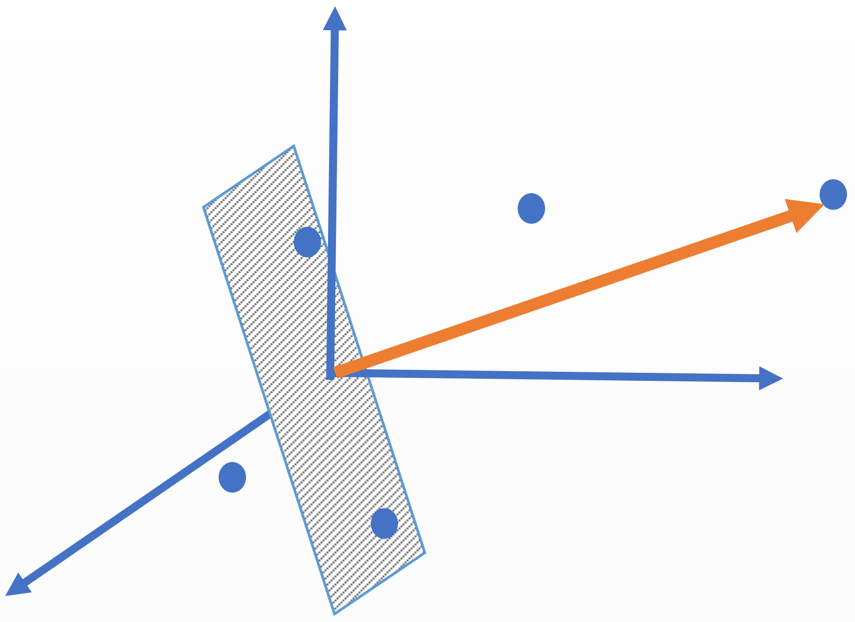
# Adaptive Sampling Example



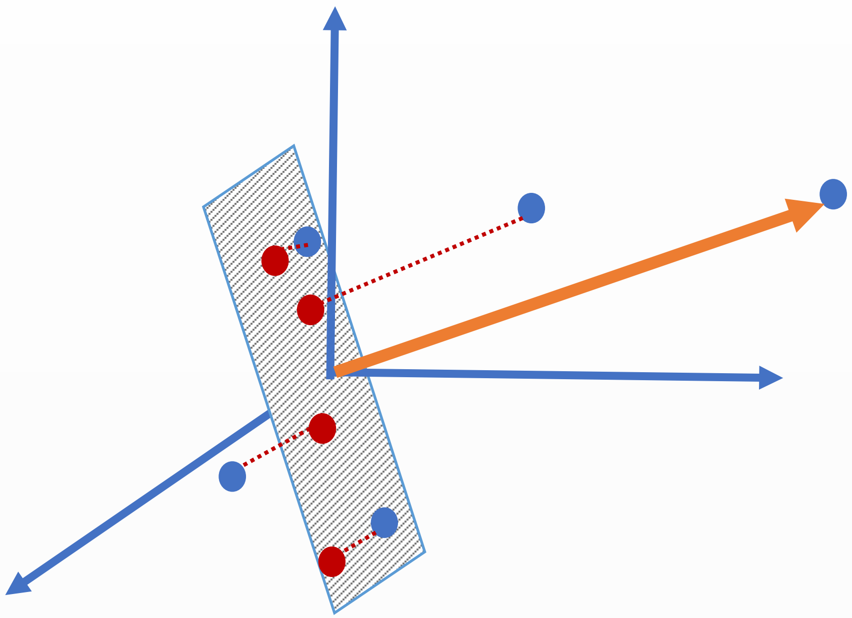
# Adaptive Sampling Example



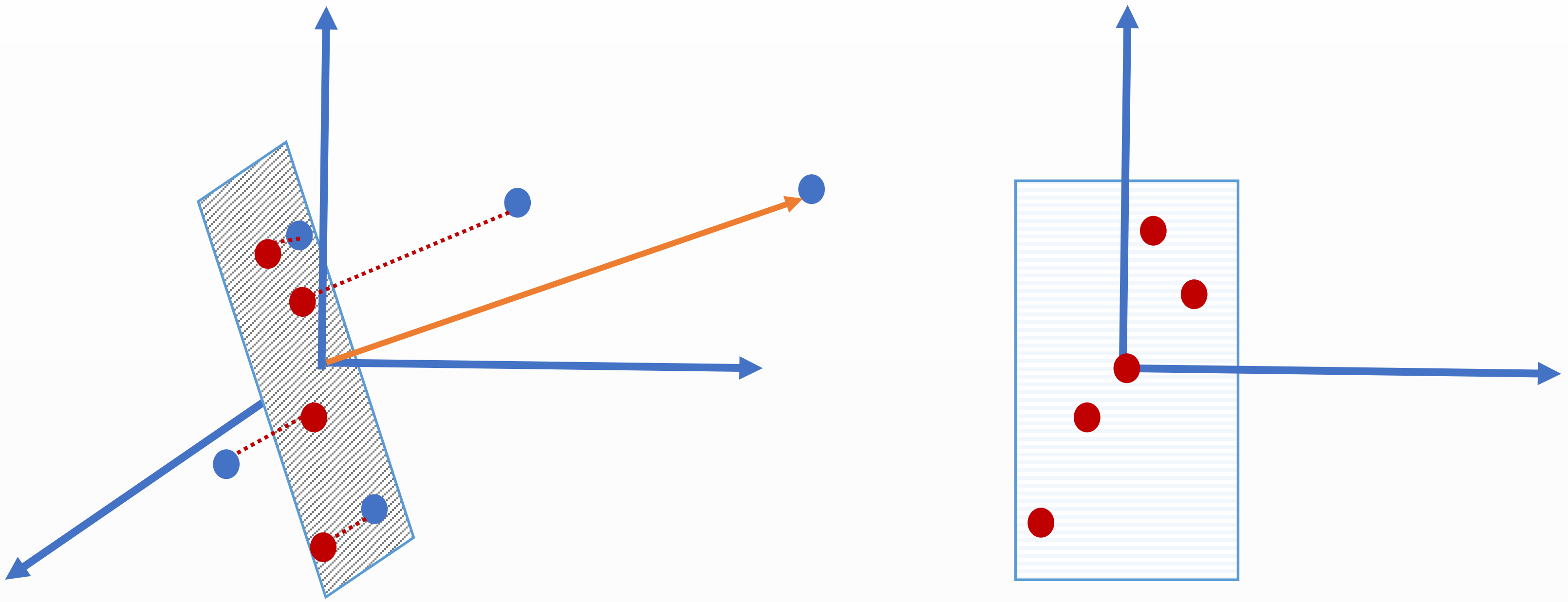
# Adaptive Sampling Example



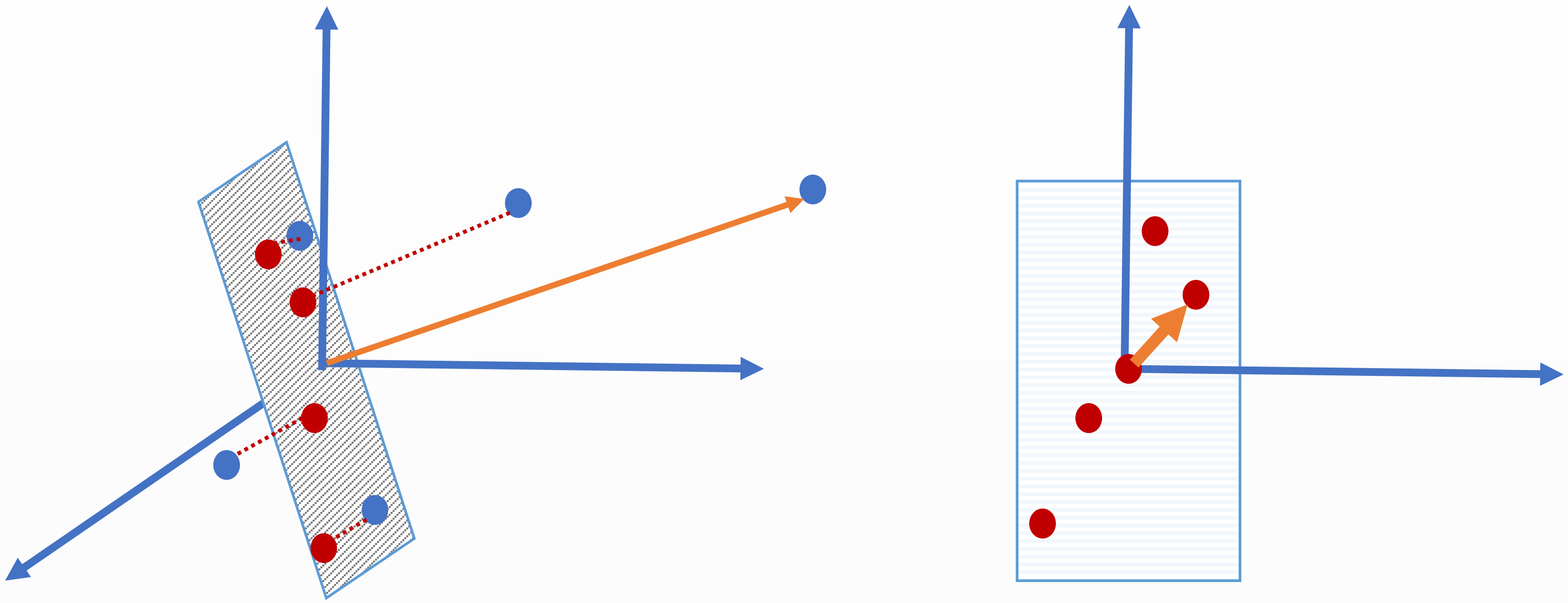
# Adaptive Sampling Example



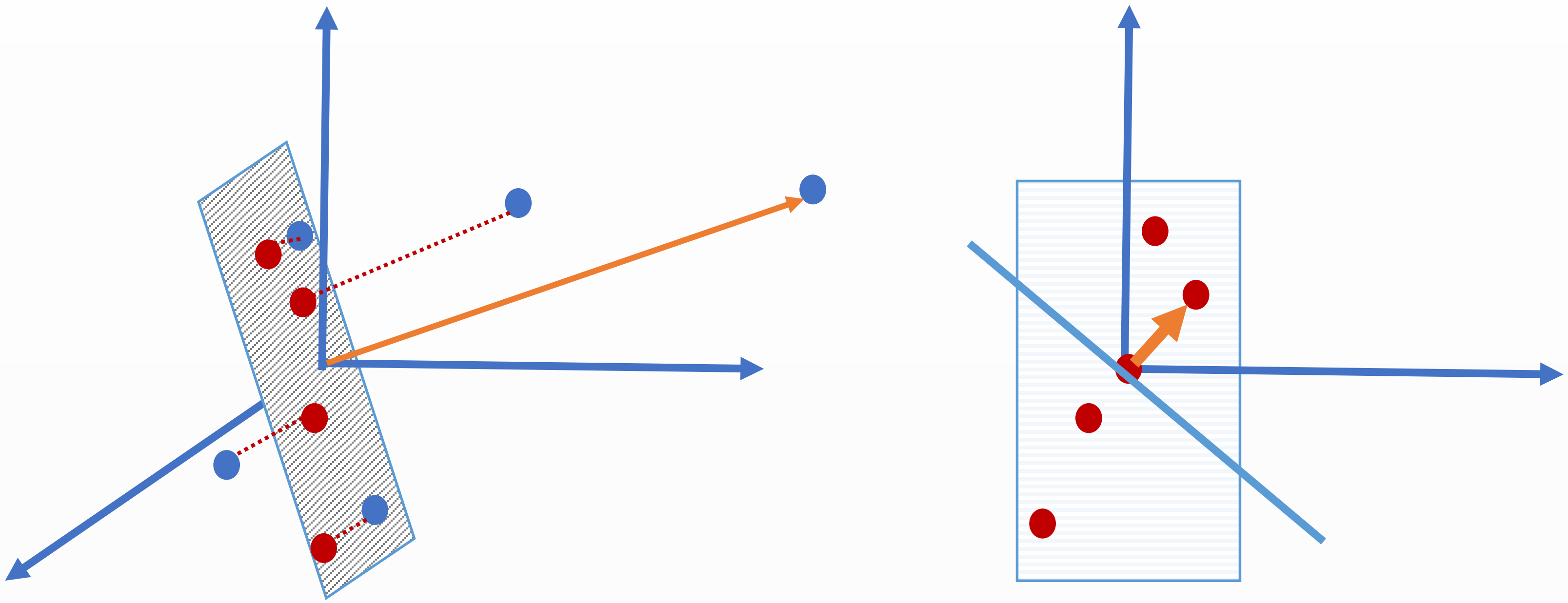
# Adaptive Sampling Example



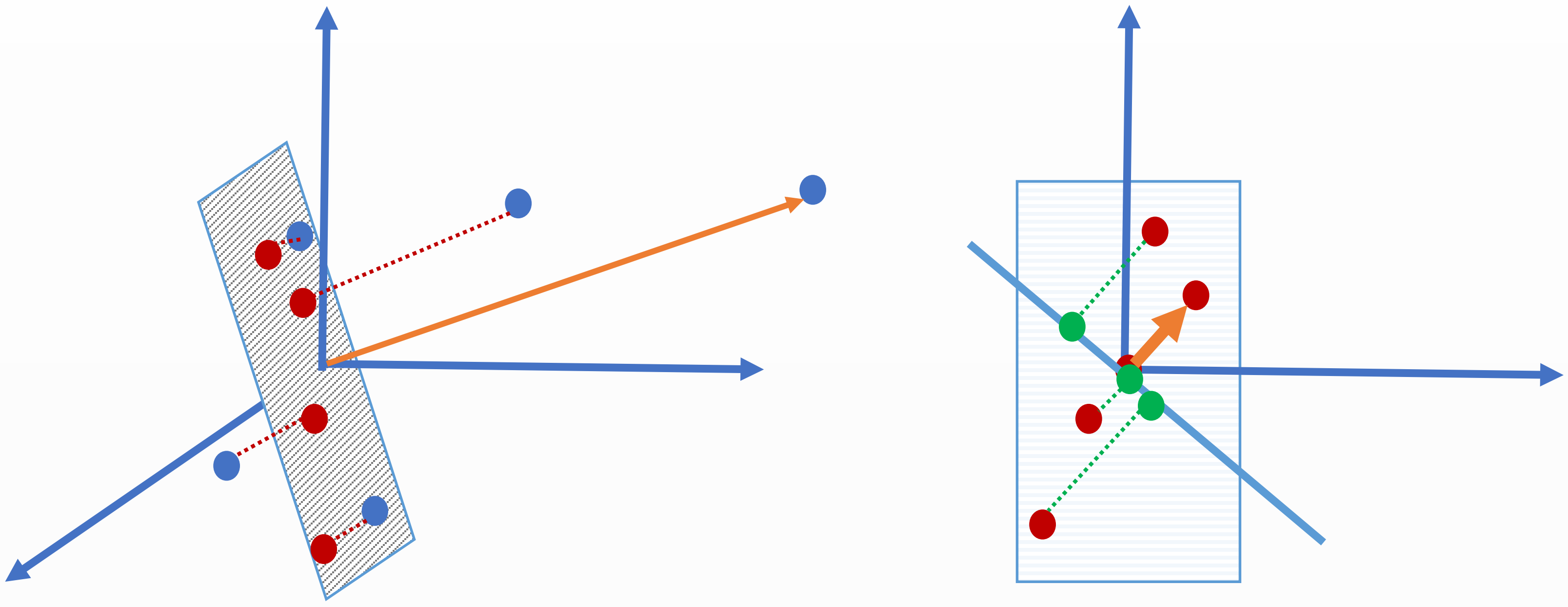
# Adaptive Sampling Example



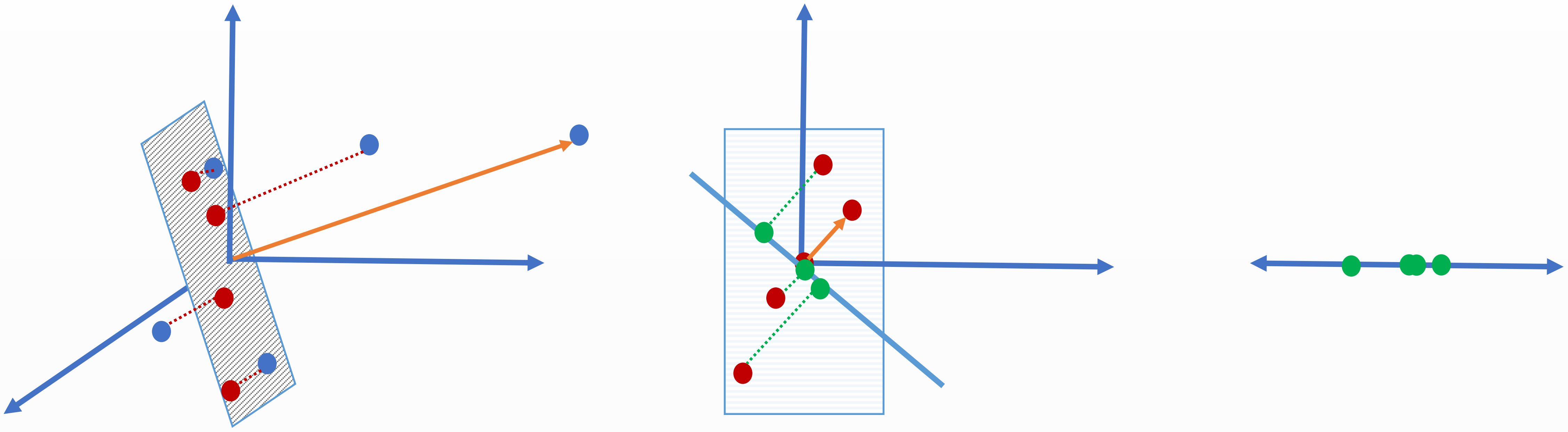
# Adaptive Sampling Example



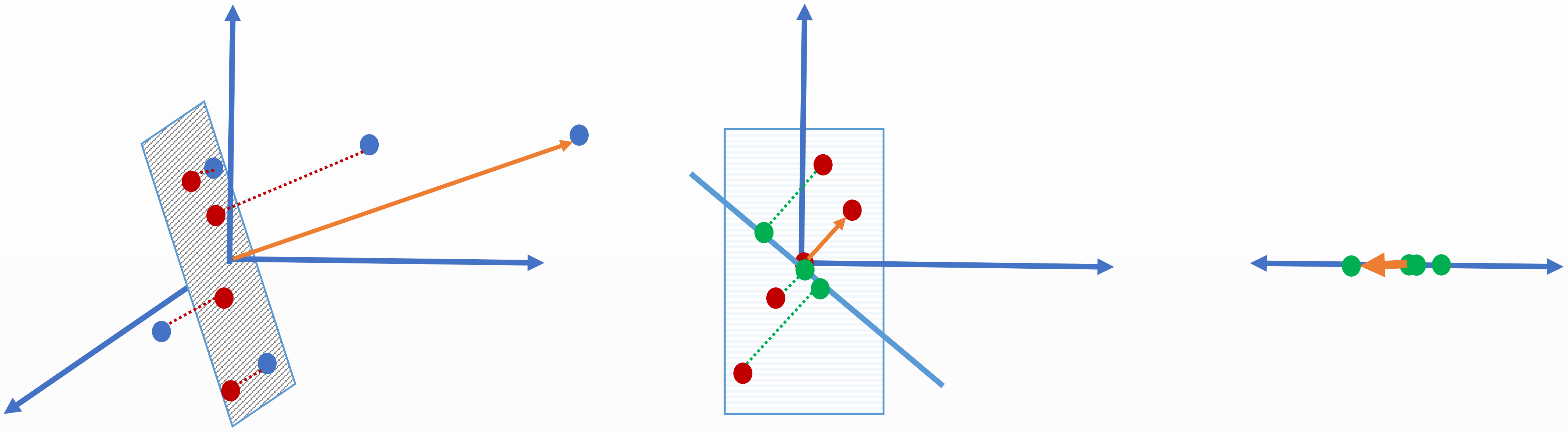
# Adaptive Sampling Example



# Adaptive Sampling Example



# Adaptive Sampling Example



# Data Summarization Tasks

**Given:**

- $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{d}}$
- parameter $k$

Rows correspond to $n$ data points

e.g. feature vectors of objects in a dataset

**Goal:**

- Find a representation (of "size $k$") for the data
- Optimize a predefined function

# Data Summarization Tasks

**Given:**

- $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n \times d}}$
- parameter $k$

**Goal:**

- Find a representation (of "size $k$") for the data
- Optimize a predefined function

**Instances:**

- **Row/Column subset selection**
- Subspace approximation
- Projective clustering
- Volume sampling/maximization

- Find a subset $S$ of $k$ rows minimizing the squared distance of all rows to the subspace of $S$

$$\|A - Proj_S(A)\|_F$$

➢ Best set of representatives

# Data Summarization Tasks

**Given:**

- $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{d}}$
- parameter $k$

**Goal:**

- Find a representation (of "size $k$") for the data
- Optimize a predefined function

**Instances:**

- Row/Column subset selection
- **Subspace approximation**
- Projective clustering
- Volume sampling/maximization

- Find a subspace $H$ of dimension $k$ minimizing the squared distance of all rows to $H$

$$\|A - Proj_H(A)\|_F$$

➢ Best approximation with a subspace

# Data Summarization Tasks

**Given:**

- $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{d}}$
- parameter $k$

**Goal:**

- Find a representation (of "size $k$") for the data
- Optimize a predefined function

**Instances:**

- Row/Column subset selection
- Subspace approximation
- Projective clustering
- Volume sampling/maximization

- Find $s$ subspaces $H_1, \dots, H_s$ each of dimension $k$ minimizing

$$\sum_{i=1}^{n} d(A_i, H)^2$$

➢ Best approximation with several subspaces

# Data Summarization Tasks

**Given:**

- $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{d}}$
- parameter $k$

**Goal:**

- Find a representation (of "size $k$") for the data
- Optimize a predefined function

**Instances:**

- Row/Column subset selection
- Subspace approximation
- Projective clustering
- **Volume sampling/maximization**

- Find a subset $S$ of $k$ rows that maximizes the volume of the parallelepiped spanned by $S$

➢ Notion for capturing diversity

➢ Maximizing diversity

# Data Summarization Tasks

**Given:**

- $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{n \times d}$
- parameter $k$

**Goal:**

- Find a representation (of "size $k$") for the data
- Optimize a predefined function

**Instances:**

- Row/Column subset selection
- Subspace approximation
- Projective clustering
- Volume sampling/maximization

Adaptive sampling is used to derive algorithms for all these tasks

# Adaptive Sampling

[DeshpandeVempala06, DeshpandeVaradarajan07, DeshpandeRademacherVempalaWang06]

- Sample row $i$ w.p. proportional to distance squared $\|A_i\|_2^2$

- **Given:** $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n} \times \boldsymbol{d}}$, parameter $k$
  - Sample a row $A_i$ with probability $\frac{\|A_i\|_2^2}{\|A\|_F^2}$

# Adaptive Sampling

[DeshpandeVempala06, DeshpandeVaradarajan07, DeshpandeRademacherVempalaWang06]

- Sample row $i$ w.p. proportional to distance squared $\|A_i\|_2^2$

- **Given:** $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n \times d}}$, parameter $k$
  - Sample a row $A_i$ with probability $\frac{\|A_i\|_2^2}{\|A\|_F^2}$

Frobenius norm:

$$\|A\|_F = \sqrt{\sum_i \sum_j A_{i,j}^2}$$

# Adaptive Sampling

**Project away** from sampled subspace

$\boldsymbol{M}^+$ :Moore-Penrose Pseudoinverse

[DeshpandeVempala06, DeshpandeVaradarajan07, DeshpandeRademacherVempalaWang06]

- Sample row $i$ w.p. proportional to $\|A_i(I - M^+M)\|_2^2$

- **Given:** $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n}\times\boldsymbol{d}}$, parameter $k$
- $M \leftarrow \emptyset$
- For $k$ rounds,
  - Sample a row $A_i$ with probability $\frac{\|A_i(I-M^+M)\|_2^2}{\|A(I-M^+M)\|_F^2}$
  - Append $A_i$ to $M$

# Adaptive Sampling

**Project away** from sampled subspace

$\boldsymbol{M}^+$ :Moore-Penrose Pseudoinverse

[DeshpandeVempala06, DeshpandeVaradarajan07, DeshpandeRademacherVempalaWang06]

- Sample row $i$ w.p. proportional to $\|A_i(I-M^+M)\|_2^2$

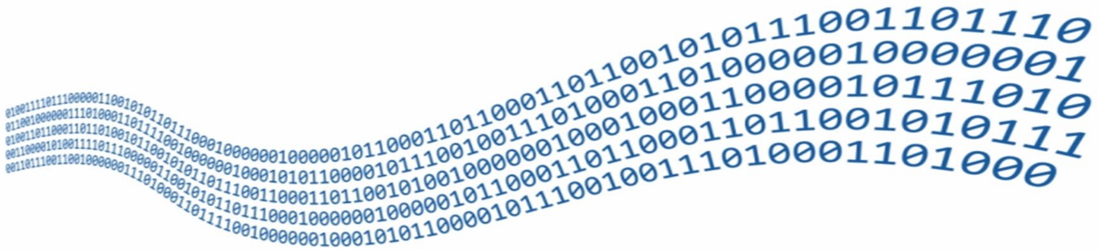
- **Given:** $n$ by $d$ matrix $\boldsymbol{A} \in \mathbb{R}^{\boldsymbol{n}\times\boldsymbol{d}}$, parameter $k$
- $M \leftarrow \emptyset$
- For $k$ rounds,
  - Sample a row $A_i$ with probability $\frac{\|A_i(I-M^+M)\|_2^2}{\|A(I-M^+M)\|_F^2}$
  - Append $A_i$ to $M$

Seems inherently sequential

**Question:**

Can we implement Adaptive Sampling in one pass (non-adaptively)?

# Streaming Algorithms



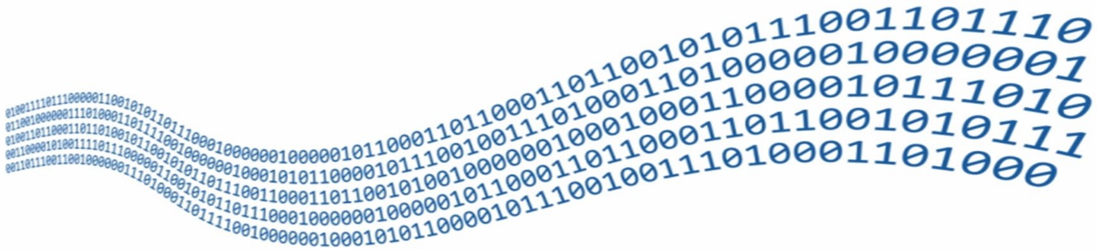
**Motivation:** Data is huge and cannot be stored in the main memory

**Streaming algorithms:** Given sequential access to the data, make one or several passes over input

- Solve the problem on the fly
- Use sub-linear storage

**Parameters**: Space, number of passes, approximation

# Streaming Algorithms



**Motivation:** Data is huge and cannot be stored in the main memory

**Streaming algorithms:** Given sequential access to the data, make one or several passes over input

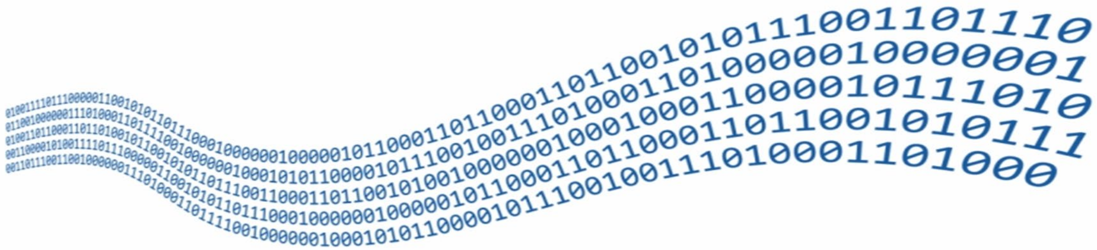
- Solve the problem on the fly
- Use sub-linear storage

**Parameters**: Space, number of passes, approximation

**Models:**

- **Row Arrival:** rows of $A$ arrive one by one
- **Turnstile:** we receive updates to the entries of the matrix i.e., $(i, j, \Delta)$ means $A_{i,j} \leftarrow A_{i,j} + \Delta$

# Streaming Algorithms



**Motivation:** Data is huge and cannot be stored in the main memory

**Streaming algorithms:** Given sequential access to the data, make one or several passes over input

- Solve the problem on the fly
- Use sub-linear storage

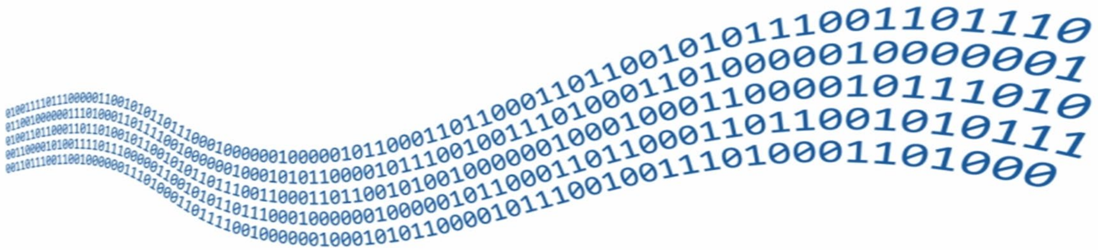
**Parameters**: Space, number of passes, approximation

**Models:**

Focus on the row arrival model for the talk

- **Row Arrival:** rows of $A$ arrive one by one
- **Turnstile:** we receive updates to the entries of the matrix i.e., $(i, j, \Delta)$ means $A_{i,j} \leftarrow A_{i,j} + \Delta$

# Streaming Algorithms



**Motivation:** Data is huge and cannot be stored in the main memory

**Streaming algorithms:** Given sequential access to the data, make one or several passes over input

- Solve the problem on the fly
- Use sub-linear storage

**Parameters**: Space, number of passes, approximation

**Models:**

Focus on the row arrival model for the talk

- **Row Arrival:** rows of $A$ arrive one by one
- **Turnstile:** we receive updates to the entries of the matrix i.e., $(i, j, \Delta)$ means $A_{i,j} \leftarrow A_{i,j} + \Delta$

**Our goal:** Simulate $k$ rounds of adaptive sampling in 1 pass of streaming

➢ Data Summarization tasks were considered in the streaming models in earlier works that used adaptive sampling [e.g. DV'06, DR'10, DRVW'06]

# Outline of Results

1. Simulate adaptive sampling in 1 pass turnstile stream
    - $L_{p,2}$ sampling with post processing matrix $P$
2. Applications in turnstile stream
    - Row/column subset selection
    - Subspace approximation
    - Projective clustering
    - Volume Maximization
3. Volume maximization lower bounds
4. Volume maximization in row arrival

# Outline of Results

1. Simulate adaptive sampling in 1 pass turnstile stream
    - **$L_{p,2}$ sampling with post processing matrix $P$**
2. Applications in turnstile stream
    - Row/column subset selection
    - Subspace approximation
    - Projective clustering
    - Volume Maximization
3. Volume maximization lower bounds
4. Volume maximization in row arrival

# Results: $\boldsymbol{L_{2,2}}$ Sampling with Post-Processing

**Input:**

- $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream
- a post-processing $\boldsymbol{P} \in \mathbb{R}^{d \times d}$

**Output:** samples an index $i \in [n]$ w.p. $\frac{\|\boldsymbol{A_i P}\|_2^2}{\|\boldsymbol{AP}\|_F^2}$

# Results: $\boldsymbol{L_{2,2}}$ Sampling with Post-Processing

**Input:**

- $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream
- a post-processing $\boldsymbol{P} \in \mathbb{R}^{d \times d}$

**Output:** samples an index $i \in [n]$ w.p. $\frac{\|\boldsymbol{A_i P}\|_2^2}{\|\boldsymbol{AP}\|_F^2}$

$\boldsymbol{P}$ corresponds to the projection matrix $(I - M^+ M)$

# Results: $\boldsymbol{L_{2,2}}$ Sampling with Post-Processing

**Input:**

- $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream
- a post-processing $\boldsymbol{P} \in \mathbb{R}^{d \times d}$

**Output:** samples an index $i \in [n]$ w.p. $(1 \pm \epsilon) \frac{\|\boldsymbol{A_i P}\|_2^2}{\|\boldsymbol{AP}\|_F^2} + \frac{1}{poly(n)}$

✓ In one pass

✓ $poly(d, \epsilon^{-1}, \log n)$ space

# Results: $L_{2,2}$ Sampling with Post-Processing

**Input:**

- $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream
- a post-processing $\boldsymbol{P} \in \mathbb{R}^{d \times d}$

**Output:** samples an index $i \in [n]$ w.p. $(1 \pm \epsilon) \frac{\|\boldsymbol{A_i P}\|_2^2}{\|\boldsymbol{AP}\|_F^2} + \frac{1}{poly(n)}$

✓ In one pass

✓ $poly(d, \epsilon^{-1}, \log n)$ space

**Impossible to return entire row instead of index in sublinear space**

❑ A long stream of *small* updates + an arbitrarily *large* update

# Results: $L_{p,2}$ Sampling with Post-Processing

**Input:**

- $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream, $\boldsymbol{p} \in \{\mathbf{1}, \mathbf{2}\}$
- a post-processing $\boldsymbol{P} \in \mathbb{R}^{d \times d}$

**Output:** samples an index $i \in [n]$ w.p. $(1 \pm \epsilon) \frac{\|\boldsymbol{A_i P}\|_2^{\boldsymbol{p}}}{\|\boldsymbol{AP}\|_{\boldsymbol{p},2}^{\boldsymbol{p}}} + \frac{1}{poly(n)}$

✓ In one pass

✓ $poly(d, \epsilon^{-1}, \log n)$ space

**Impossible to return entire row instead of index in sublinear space**

❑ A long stream of *small* updates + an arbitrarily *large* update

# Outline of Results

1. **Simulate adaptive sampling in 1 pass turnstile stream**
   - $L_{p,2}$ sampling with post processing matrix $P$
2. Applications in turnstile stream
   - Row/column subset selection
   - Subspace approximation
   - Projective clustering
   - Volume Maximization
3. Volume maximization lower bounds
4. Volume maximization in row arrival

# Results: Adaptive Sampling

**Input:** $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream

**Output:** Return each set $\boldsymbol{S} \subset_{\boldsymbol{k}} [\boldsymbol{n}]$ of $k$ indices w.p. $\boldsymbol{p_S}$ s.t.

$\sum_S |\boldsymbol{p_S} - \boldsymbol{q_S}| \leq \epsilon$

- $\boldsymbol{q_S}$: prob. of selecting $\boldsymbol{S}$ via **adaptive sampling**
- w.r.t. either distance or squared distance (i.e., $p \in \{1,2\}$)

# Results: Adaptive Sampling

**Input:** $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream

**Output:** Return each set $\boldsymbol{S} \subset_{\boldsymbol{k}} [\boldsymbol{n}]$ of $k$ indices w.p. $\boldsymbol{p_S}$ s.t.

$$\sum_S |\boldsymbol{p_S} - \boldsymbol{q_S}| \le \epsilon$$

- $\boldsymbol{q_S}$: prob. of selecting $\boldsymbol{S}$ via **adaptive sampling**
- w.r.t. either distance or squared distance (i.e., $p \in \{1,2\}$)

✓ In one pass

✓ $poly(d, k, \epsilon^{-1}, \log n)$ space

# Results: Adaptive Sampling

**Input:** $A \in \mathbb{R}^{n\times d}$ as a (turnstile) stream

**Output:** Return each set $\boldsymbol{S} \subset_k [\boldsymbol{n}]$ of $k$ indices w.p. $\boldsymbol{p_S}$ s.t.

$\sum_S |\boldsymbol{p_S} - \boldsymbol{q_S}| \le \epsilon$

- $\boldsymbol{q_S}$: prob. of selecting $\boldsymbol{S}$ via **adaptive sampling**
- w.r.t. either distance or squared distance (i.e., $p \in \{1,2\}$)

✓ In one pass

✓ $poly(d, k, \epsilon^{-1}, \log n)$ space

✓ Besides indices S, a noisy set of rows $r_1, \ldots, r_k$ are returned

- Each $r_i$ is close to the corresponding $A_i$ (w.r.t. residual)

# Results: Adaptive Sampling

**Input:** $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream

**Output:** Return each set $\boldsymbol{S} \subset_k [\boldsymbol{n}]$ of $k$ indices w.p. $\boldsymbol{p_S}$ s.t.

$$\sum_S |\boldsymbol{p_S} - \boldsymbol{q_S}| \leq \epsilon$$

- $\boldsymbol{q_S}$: prob. of selecting $\boldsymbol{S}$ via **adaptive sampling**
- w.r.t. either distance or squared distance (i.e., $p \in \{1,2\}$)

✓ In one pass

✓ $poly(d, k, \epsilon^{-1}, \log n)$ space

✓ Besides indices S, a noisy set of rows $r_1, \ldots, r_k$ are returned

- Each $r_i$ is close to the corresponding $A_i$ (w.r.t. residual)

Impossible to return the row accurately in sublinear space

- ❑ A long stream of *small* updates + an arbitrarily *large* update

# Outline of Results

1. Simulate adaptive sampling in 1 pass turnstile stream
    - $L_{p,2}$ sampling with post processing matrix $P$

2. **Applications in turnstile stream**
    - Row/column subset selection
    - Subspace approximation
    - Projective clustering
    - Volume Maximization

3. Volume maximization lower bounds

4. Volume maximization in row arrival

# Applications: Row Subset Selection

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$ rows of $A$ to form $\boldsymbol{M}$ to minimize $||A - AM^{+}M||_F$

# Applications: Row Subset Selection

**Input:** $A \in \mathbb{R}^{n\times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$ rows of $A$ to form $\boldsymbol{M}$ to minimize $\|A - AM^{+}M\|_F$

**Our Result:** finds **M** such that,

$$\mathbf{Pr}[\|A - A\boldsymbol{M}^{+}\boldsymbol{M}\|_F^2 \leq \mathbf{16}(\boldsymbol{k}+\mathbf{1})!\,\|A - A_k\|_F^2] \geq 2/3$$

- $A_k$: best rank-$k$ approximation of $A$
- first one pass turnstile streaming algorithm
- $poly(d, k, \log n)$ space

# Applications: Row Subset Selection

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$ rows of $A$ to form $\boldsymbol{M}$ to minimize $\|A - AM^{+}M\|_F$

**Our Result:** finds **M** such that,

$$\mathbf{Pr}[\|A - A\boldsymbol{M}^{+}\boldsymbol{M}\|_F^2 \leq \mathbf{16}(\boldsymbol{k} + \mathbf{1})!\,\|A - A_k\|_F^2] \geq 2/3$$

- $A_k$: best rank-$k$ approximation of $A$
- first one pass turnstile streaming algorithm
- $poly(d, k, \log n)$ space

➢ Previous works: centralized setting **[e.g. DRVW06, BMD09, GS'12]** and row arrival **[e.g., CMM'17, GP'14 , BDMMUWZ'18]**

# Applications: Subspace Approximation

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$-dim subspace $\boldsymbol{H}$ to minimize $\left(\sum_{i=1}^{n} d(A_i, H)^p\right)^{1/p}$

- $p \in \{1,2\}$
- $d(A_i, H) = \|A_i(\mathbb{I} - H^+ H)\|_2$

# Applications: Subspace Approximation

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$-dim subspace $\boldsymbol{H}$ to minimize $\left(\sum_{i=1}^{n} d(A_i, H)^p\right)^{1/p}$

- $p \in \{1,2\}$
- $d(A_i, H) = \|A_i(\mathbb{I} - H^+ H)\|_2$

**Our Result I:** finds **H** (which is ***k* noisy rows of *A***) s.t.,

$$\mathbf{Pr}\left[\left(\sum_{i=1}^{n} d(A_i, \boldsymbol{H})^p\right)^{1/p} \leq \mathbf{4}(\boldsymbol{k} + \mathbf{1})! \left(\sum_{i=1}^{n} d(A_i, A_k)^p\right)^{1/p}\right] \geq \frac{2}{3}$$

- $A_k$: best rank-*k* approximation of A
- $poly(d, k, \log n)$ space

# Applications: Subspace Approximation

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$-dim subspace $\boldsymbol{H}$ to minimize $(\sum_{i=1}^{n} d(A_i, H)^p)^{1/p}$

- $p \in \{1,2\}$
- $d(A_i, H) = \|A_i(\mathbb{I} - H^+H)\|_2$

**Our Result I:** finds **H** (which is ***k* noisy rows of *A***) s.t.,

$$\mathbf{Pr}[(\sum_{i=1}^{n} d(A_i, \boldsymbol{H})^p)^{1/p} \leq \mathbf{4}(\boldsymbol{k}+\mathbf{1})!\,(\sum_{i=1}^{n} d(A_i, A_k)^p)^{1/p}] \geq \frac{2}{3}$$

- $A_k$: best rank-$k$ approximation of A
- $poly(d, k, \log n)$ space
- First relative error on turnstile streams that returns noisy rows of A

➢ **[Levin, Sevekari, Woodruff'18]**

+$(1+\epsilon)$-approximation –larger number of rows –rows are not from A

# Applications: Subspace Approximation

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$-dim subspace $\boldsymbol{H}$ to minimize $\left(\sum_{i=1}^{n} d(A_i, H)^p\right)^{1/p}$

- $p \in \{1,2\}$
- $d(A_i, H) = \|A_i(\mathbb{I} - H^+ H)\|_2$

**Our Result II:** finds **H** (which is $\boldsymbol{poly(k, 1/\epsilon)}$ ***noisy* rows of *A***) s.t.,

$$\mathbf{Pr}\left[\left(\sum_{i=1}^{n} d(A_i, \boldsymbol{H})^p\right)^{1/p} \le (\mathbf{1} + \boldsymbol{\epsilon})\left(\sum_{i=1}^{n} d(A_i, A_k)^p\right)^{1/p}\right] \ge \frac{2}{3}$$

- $A_k$: best rank-*k* approximation of A
- $\boldsymbol{poly(d, k, 1/\epsilon, \log n)}$ space

➢ **[Levin, Sevekari, Woodruff'18]**

–$poly(\log(nd), k, 1/\epsilon)$ rows –rows are not from A

# Applications: Projective Clustering

**Input:** $A \in \mathbb{R}^{n \times d}$, target dim $\boldsymbol{k}$ and target number of subspaces $\boldsymbol{s}$

**Output:** $\boldsymbol{s}$ $\boldsymbol{k}$-dim subspaces $\boldsymbol{H_1}, \ldots, \boldsymbol{H_s}$ to minimize $\left(\sum_{i=1}^{n} d(A_i, \boldsymbol{H})^p\right)^{1/p}$

- $\boldsymbol{H} = \boldsymbol{H_1} \cup \cdots \cup \boldsymbol{H_s}$ and $p \in \{1,2\}$
- $d(A_i, H) = \min_{j \in [s]} \left\| A_i \left( \mathbb{I} - H_j^{+} H_j \right) \right\|_2$

# Applications: Projective Clustering

**Input:** $A \in \mathbb{R}^{n \times d}$, target dim $\boldsymbol{k}$ and target number of subspaces $\boldsymbol{s}$

**Output:** $\boldsymbol{s}$ $\boldsymbol{k}$-dim subspaces $\boldsymbol{H}_1, \dots, \boldsymbol{H}_s$ to minimize $\left(\sum_{i=1}^{n} d(A_i, \boldsymbol{H})^p\right)^{1/p}$

- $\boldsymbol{H} = \boldsymbol{H}_1 \cup \cdots \cup \boldsymbol{H}_s$ and $p \in \{1,2\}$
- $d(A_i, H) = \min_{j \in [s]} \left\| A_i \left( \mathbb{I} - H_j^+ H_j \right) \right\|_2$

**Our Result:** finds **S** (which is $\boldsymbol{poly(k, s, 1/\epsilon)}$ ***noisy* rows of $\boldsymbol{A}$**),

which contains a union **T** of $s$ $\boldsymbol{k}$-dim subspaces s.t.,

$$\mathbf{Pr}\left[\left(\sum_{i=1}^{n} d(A_i, \boldsymbol{T})^p\right)^{1/p} \le (\mathbf{1} + \boldsymbol{\epsilon})\left(\sum_{i=1}^{n} d(A_i, \boldsymbol{H})^p\right)^{1/p}\right] \ge 2/3$$

- $H$: optimal solution to projective clustering
- first one pass turnstile streaming algorithm with sublinear space
- $poly(d, k, \log n, s, 1/\epsilon)$ space

➢ **[BHI'02, HM'04, Che09, FMSW'10]** based on coresets, works in row arrival

➢ **[KR'15]** turnstile but linear in number of points

# Applications: Volume Maximization

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$

**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

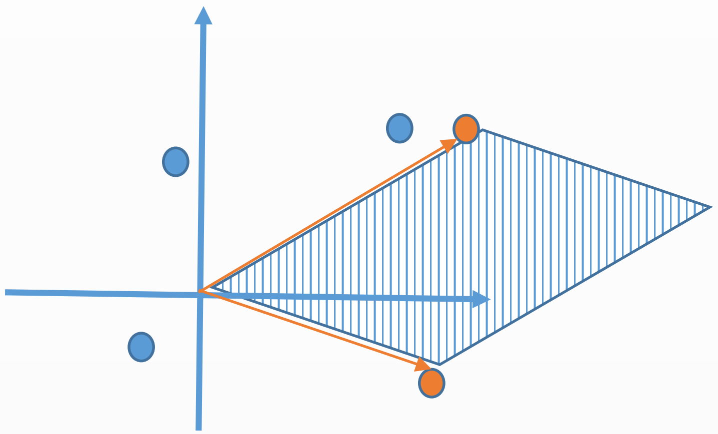
# Applications: Volume Maximization

Volume of the parallelepiped spanned by those vectors

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$

**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

$\boldsymbol{k = 2}$



# Applications: Volume Maximization

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

**Our Result (Upper Bound I):** for an approximation factor $\boldsymbol{\alpha}$, finds **S** (set of $\boldsymbol{k}$ ***noisy* rows of $\boldsymbol{A}$**) s.t.,

$$\mathbf{Pr}[\alpha^k (k!) \mathrm{Vol}(\mathbf{S}) \geq \mathrm{Vol}(\mathbf{M})] \geq 2/3$$

- first one pass turnstile streaming algorithm
- $\tilde{O}(ndk^2/\alpha^2)$ space

# Applications: Volume Maximization

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

**Our Result (Upper Bound I):** for an approximation factor $\boldsymbol{\alpha}$, finds **S** (set of ***$k$ noisy rows of $A$***) s.t.,

$$\mathbf{Pr}[\alpha^k (k!) \mathrm{Vol}(\mathbf{S}) \geq \mathrm{Vol}(\mathbf{M})] \geq 2/3$$

- first one pass turnstile streaming algorithm
- $\tilde{O}(ndk^2/\alpha^2)$ space

➢ [Indyk, **M**, Oveis Gharan, Rezaei, '19 '20] coreset based $\tilde{O}(k)^{k/\epsilon}$ approx. and $\tilde{O}(n^\epsilon kd)$ space for row-arrival streams

# Outline of Results

1. Simulate adaptive sampling in 1 pass turnstile stream
   - $L_{p,2}$ sampling with post processing matrix $P$
2. Applications in turnstile stream
   - Row/column subset selection
   - Subspace approximation
   - Projective clustering
   - Volume Maximization
3. **Volume maximization lower bounds**
4. **Volume maximization in row arrival**

# Volume Maximization Lower Bounds

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

**Our Result (Lower Bound I):** for $\boldsymbol{\alpha}$, any $\boldsymbol{p}$-pass algorithm that finds $\boldsymbol{\alpha}^k$-approximation w.p. $\geq 63/64$ in **turnstile-arrival** requires $\Omega(n/k\boldsymbol{p}\boldsymbol{\alpha}^2)$ space.

- Our previous upper bound is matches the upper bound up to a factor of $\boldsymbol{k^3 d}$ in space and $\boldsymbol{k}!$ in the approximation factor.

# Volume Maximization Lower Bounds

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \dots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

**Our Result (Lower Bound II):** for a fixed constant $\boldsymbol{C}$, any one-pass algorithm that finds $\boldsymbol{C^k}$-approximation w.p. $\geq 63/64$ in **random order row-arrival** requires $\Omega(n)$ space

# Volume Maximization – Row Arrival

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

**Our Result (Upper Bound II):** for an approximation factor $\boldsymbol{C} < (\log n)/k$, finds **S** (set of **$\boldsymbol{k}$ rows of $\boldsymbol{A}$**) s.t.

- approximation factor $\tilde{O}(\boldsymbol{C}k)^{k/2}$ with high probability
- one pass **row-arrival** streaming algorithm
- $\tilde{O}(n^{O(1/\boldsymbol{C})} d)$ space

# Volume Maximization – Row Arrival

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

**Our Result (Upper Bound II):** for an approximation factor $\boldsymbol{C} < (\log n)/k$, finds **S** (set of **$k$ rows of $A$**) s.t.

- approximation factor $\tilde{O}(\boldsymbol{C}k)^{k/2}$ with high probability
- one pass **row-arrival** streaming algorithm
- $\tilde{O}(n^{O(1/\boldsymbol{C})}d)$ space

➢ [Indyk, **M**, Oveis Gharan, Rezaei, '19 '20] coreset based $\tilde{O}(k)^{\boldsymbol{C}k/2}$ approx. and $\tilde{O}(n^{1/\boldsymbol{C}}kd)$ space for row-arrival streams

# $L_{p,2}$ Sampler

1. Simulate adaptive sampling in 1 pass
   - $L_{p,2}$ sampling with post processing matrix $P$
2. Applications in turnstile stream
   - Row/column subset selection
   - Subspace approximation
   - Projective clustering
   - Volume Maximization
3. Volume maximization lower bounds
4. Volume maximization in row arrival

# $L_{2,2}$ Sampler with Post-Processing Matrix

**Input:** matrix **A** as a data stream, a post-processing matrix ***P***

**Output:** index $i$ of a row of **AP** sampled w.p. $\sim \frac{\|A_i P\|_2^2}{\|AP\|_F^2}$

# $L_{2,2}$ Sampler with Post-Processing Matrix

**Input:** matrix **A** as a data stream, a post-processing matrix ***P***

**Output:** index $i$ of a row of **AP** sampled w.p. $\sim \frac{\|A_i P\|_2^2}{\|AP\|_F^2}$

**Extension of $L_2$ Sampler**

[Andoni et al.'10][Monemizadeh, Woodruff'10][Jowhari et al.'11][Jayaram, Woodruff'18]

**Input:** vector **f** as a data stream

**Output:** index $i$ of a coordinate of **f** sampled w.p. $\sim \frac{f_i^2}{\|\mathbf{f}\|_2^2}$

# $L_{2,2}$ Sampler with Post-Processing Matrix

**Input:** matrix **A** as a data stream, a post-processing matrix ***P***

**Output:** index $i$ of a row of **AP** sampled w.p. $\sim \frac{\|A_i P\|_2^2}{\|AP\|_F^2}$

**What is new:**

1. Generalizing vectors to matrices
2. Handling the post processing matrix $P$

**Extension of $L_2$ Sampler**

[Andoni et al.'10][Monemizadeh, Wood ... druff'18]

**Input:** vector **f** as a data stream

**Output:** index $i$ of a coordinate of **f** sampled w.p. $\sim \frac{f_i^2}{\|\mathbf{f}\|_2^2}$

# $L_{2,2}$ Sampler

**Input:** matrix **A** as a data stream

**Output:** index $i$ of a row of **A** sampled w.p. $\sim \frac{\|A_i\|_2^2}{\|\mathbf{A}\|_F^2}$

Ignore $P$ for now

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

$$\mathbf{Pr}[\|B_i\|_2^2 \geq \|A\|_F^2] = \mathbf{Pr}[\frac{\|A_i\|_2^2}{\|A\|_F^2} \geq t_i] = \frac{\|A_i\|_2^2}{\|A\|_F^2}$$

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

**Pr**$[\|B_i\|_2^2 \geq \|A\|_F^2]$ = **Pr**$[\frac{\|A_i\|_2^2}{\|A\|_F^2} \geq t_i]$ = $\frac{\|A_i\|_2^2}{\|A\|_F^2}$

- ❑ Return $i$ that satisfies $\|B_i\|_2^2 \geq \|A\|_F^2$

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{\mathbf{1}}{\sqrt{\boldsymbol{t_i}}} \times A_i$

$$\mathbf{Pr}[\|B_i\|_2^2 \geq \|A\|_F^2] = \mathbf{Pr}[\frac{\|A_i\|_2^2}{\|A\|_F^2} \geq t_i] = \frac{\|\boldsymbol{A_i}\|_2^2}{\|\boldsymbol{A}\|_F^2}$$

❑ Return $i$ that satisfies $\|\boldsymbol{B_i}\|_2^2 \geq \|\boldsymbol{A}\|_F^2$

**Issues:**

1. Multiple rows passing the threshold
2. Don't have access to exact values of $\|\boldsymbol{B_i}\|_2^2$ and $\|\boldsymbol{A}\|_F^2$

# $L_{2,2}$ Sampler

**Issue 1**: Multiple rows passing the threshold

- **Set the threshold higher**

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

- Ideally, return the only $\boldsymbol{i}$ that satisfies $\|\boldsymbol{B_i}\|_2^2 \geq \|\boldsymbol{A}\|_F^2$

$$\mathbf{Pr}[\|B_i\|_2^2 \geq \gamma^2 \cdot \|A\|_F^2] = \frac{1}{\gamma^2} \times \frac{\|A_i\|_2^2}{\|A\|_F^2}$$

$$\gamma^2 := \frac{C \ \log n}{\epsilon}$$

# $L_{2,2}$ Sampler

**Issue 1**: Multiple rows passing the threshold

➢ **Set the threshold higher**

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

❑ Ideally, return the only $i$ that satisfies $\|B_i\|_2^2 \geq \|A\|_F^2$

$\textbf{Pr}[\|B_i\|_2^2 \geq \gamma^2 \cdot \|A\|_F^2] = \frac{1}{\gamma^2} \times \frac{\|A_i\|_2^2}{\|A\|_F^2}$

$\gamma^2 := \frac{C \log n}{\epsilon}$

Success prob: $\Omega(\frac{\epsilon}{\log n})$

**Pr**[squared norm of at least one row exceeds $\gamma^2 \cdot \|A\|_F^2$] = $\Omega(\frac{1}{\gamma^2})$

**Pr**[squared norms of more than one row exceed $\gamma^2 \cdot \|A\|_F^2$] = $O(\frac{1}{\gamma^4})$

# $L_{2,2}$ Sampler

**Issue 1**: Multiple rows passing the threshold

- **Set the threshold higher**

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

❑ Ideally, return the only $\boldsymbol{i}$ that satisfies $\|\boldsymbol{B_i}\|_2^2 \geq \|\boldsymbol{A}\|_F^2$

$$\mathbf{Pr}[\|B_i\|_2^2 \geq \gamma^2 \cdot \|A\|_F^2] = \frac{1}{\gamma^2} \times \frac{\|A_i\|_2^2}{\|A\|_F^2}$$

$$\gamma^2 := \frac{C \log n}{\epsilon}$$

Success prob: $\Omega(\frac{\epsilon}{\log n})$

To succeed, repeat $\tilde{O}(1/\epsilon)$

# $L_{2,2}$ Sampler

**Issue 2**: Don't have access to exact values of $\|B_i\|$ and $\|A\|_F$

- **estimate $\|B_i\|_2$ and $\|A\|_F$**

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$
- Return $i$ that satisfies $\|B_i\|_2 \geq \gamma \cdot \|A\|_F$

# $L_{2,2}$ Sampler

**Issue 2**: Don't have access to exact values of $\|B_i\|$ and $\|A\|_F$

➢ **estimate $\|B_i\|_2$ and $\|A\|_F$**

**Find heaviest row using CountSketch**

**Estimate norm of A using AMS**

**Step 1.**

- pick $t_i \in [0,1]$ unifor
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

❑ Return $i$ that satisfies $\|B_i\|_2 \geq \gamma \cdot \|A\|_F$

# Count Sketch

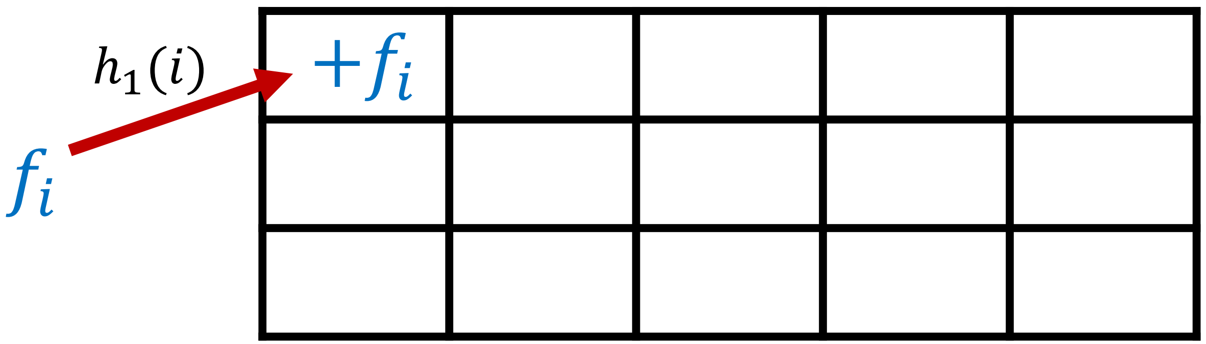
Estimate $\|\boldsymbol{B_i}\|_2$ for rows with large norms

# Count Sketch

Given a stream of items, estimate frequency of each item (i.e., coordinates in a vector)

**#rows** r $= O(\log n)$
**#buckets**/row b $= O(1/\epsilon^2)$

- **Hash** $h_j : [n] \to [b]$
- **Sign** $\sigma_j : [n] \to \{-1, +1\}$



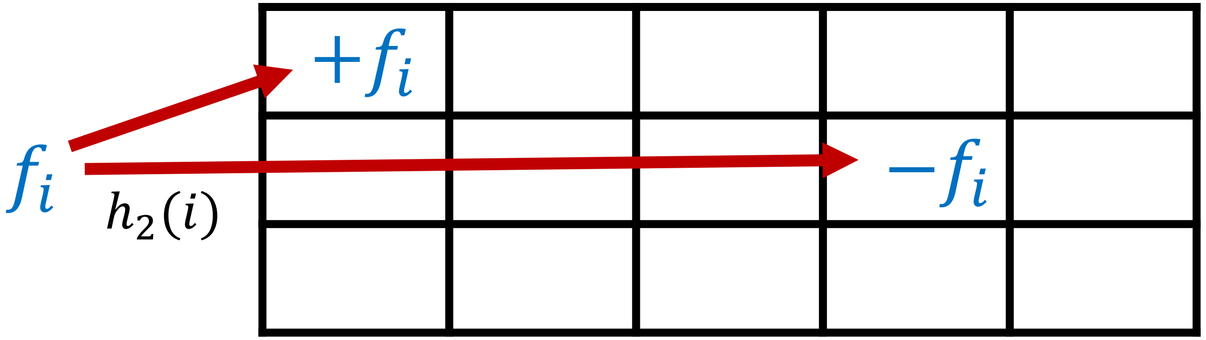

- **Update:** $C[j, h_j(i)] \mathrel{+}= \sigma_j(i) \cdot f_i$
-

# Count Sketch

Given a stream of items, estimate frequency of each item (i.e., coordinates in a vector)

**#rows** r $= O(\log n)$
**#buckets**/row b = $O(1/\epsilon^2)$

- **Hash** $h_j: [n] \to [b]$
- **Sign** $\sigma_j: [n] \to \{-1, +1\}$



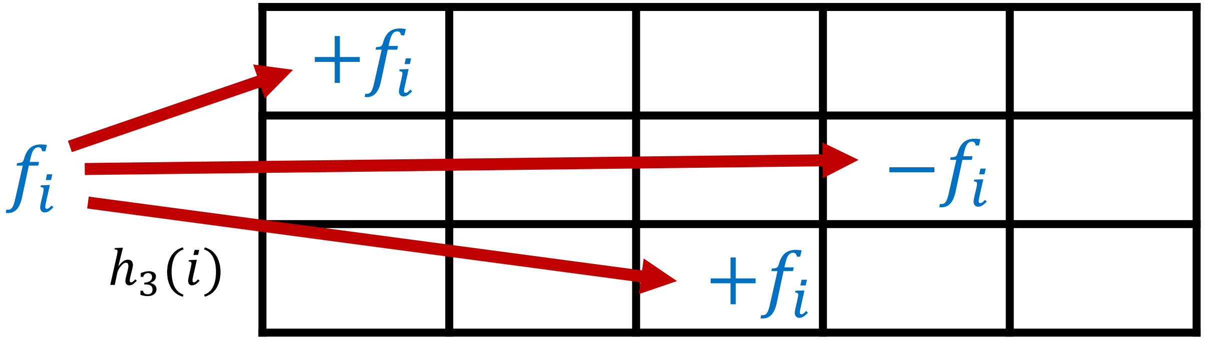

- **Update:** $C[j, h_j(i)] \mathrel{+}= \sigma_j(i) \cdot f_i$
-

# Count Sketch

Given a stream of items, estimate frequency of each item (i.e., coordinates in a vector)

**#rows** r $= O(\log n)$
**#buckets**/row b $= O(1/\epsilon^2)$

- **Hash** $h_j : [n] \to [b]$
- **Sign** $\sigma_j : [n] \to \{-1, +1\}$



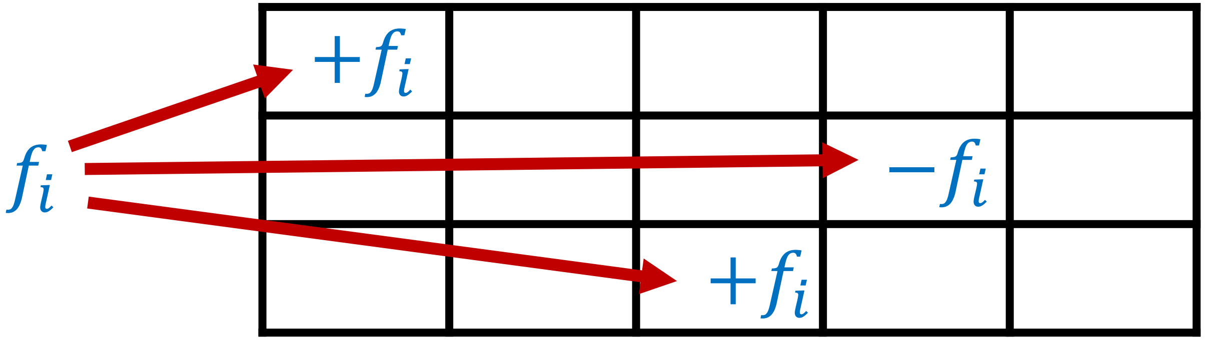

- **Update:** $C[j, h_j(i)] \mathrel{+}= \sigma_j(i) \cdot f_i$
- 

# Count Sketch

Given a stream of items, estimate frequency of each item (i.e., coordinates in a vector)

**#rows** r $= O(\log n)$
**#buckets**/row b $= O(1/\epsilon^2)$

- **Hash** $h_j : [n] \to [b]$
- **Sign** $\sigma_j : [n] \to \{-1, +1\}$

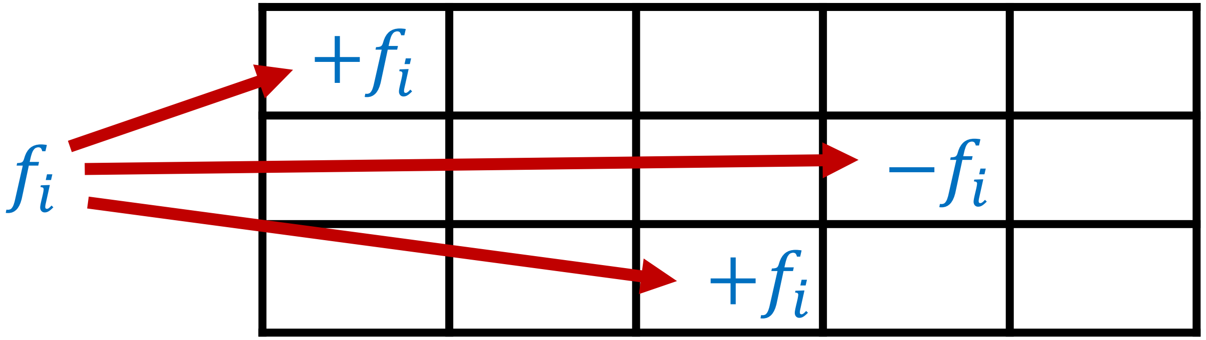



- **Update:** $C[j, h_j(i)] \mathrel{+}= \sigma_j(i) \cdot f_i$
- **Estimate** $\hat{f}_i \coloneqq median_j \; \sigma_j C[j, h_j(i)]$

# Count Sketch

Given a stream of items, estimate frequency of each item (i.e., coordinates in a vector)

**#rows** r $= O(\log n)$
**#buckets**/row b $= O(1/\epsilon^2)$




**Estimation guarantee**

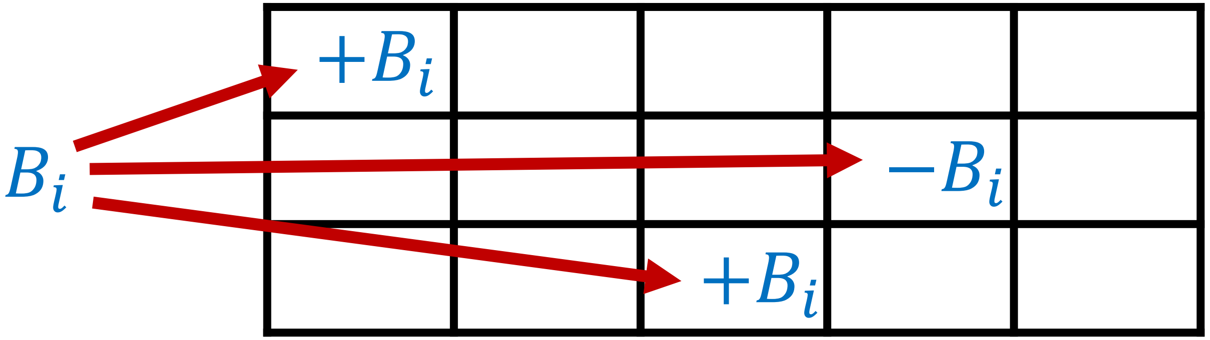
$$\left|f_i - \hat{f}_i\right| \leq \epsilon \cdot \|\mathbf{f}\|_2$$

- **Update:** $C[j, h_j(i)] \mathrel{+}= \sigma_j(i) \cdot f_i$
- **Estimate** $\hat{f}_i \coloneqq median_j\ \sigma_j C[j, h_j(i)]$

# Count Sketch

Estimate $\|\boldsymbol{B_i}\|_2$ for rows with large norms

**#rows** r $= O(\log n)$
**#buckets**/row b = $O(1/\epsilon^2)$




**Estimation guarantee**

$$\left| \|B_i\|_2 - \left\|\widehat{B}_i\right\|_2 \right| \leq \epsilon \cdot \|B\|_F$$

**Space usage:**

$$O\left(\log n \times \frac{1}{\epsilon^2}\right) \times d$$

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

**Goal:** $\|B_i\|_2 \geq \gamma \cdot \|A\|_F$

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{1}{\sqrt{t_i}} \times A_i$

**Goal:** $\|B_i\|_2 \geq \gamma \cdot \|A\|_F$

**Step 2.**

- $\|\widehat{B_i}\|_2$ is an estimate of $\|B_i\|_2$ by modified **Countsketch**
- $\widehat{F}$ is an estimate of $\|A\|_F$ by modified **AMS**

**Test:** $\|\widehat{B}_i\|_2 \geq \gamma \cdot \widehat{F}$

# $L_{2,2}$ Sampler

**Step 1.**

- pick $t_i \in [0,1]$ uniformly at random
- set $B_i := \frac{\mathbf{1}}{\sqrt{t_i}} \times A_i$

**Goal:** $\|B_i\|_2 \geq \gamma \cdot \|A\|_F$

**Step 2.**

- $\|\widehat{B_i}\|_2$ is an estimate of $\|B_i\|_2$ by modified **Countsketch**
- $\widehat{F}$ is an estimate of $\|A\|_F$ by modified **AMS**

**Test:** $\|\widehat{B}_i\|_2 \geq \gamma \cdot \widehat{F}$

➢ The test succeeds w.p. $\epsilon$, the estimate of largest row exceeds the threshold

# Handling Post-Processing Matrix

**Input:** matrix **A** as a data stream, a post-processing matrix ***P***

**Output:** index $i$ of a row of **AP** sampled w.p. $\sim \frac{\|A_i P\|_2^2}{\|AP\|_F^2}$

# Handling Post-Processing Matrix

**Input:** matrix **A** as a data stream, a post-processing matrix ***P***

**Output:** index $i$ of a row of **AP** sampled w.p. $\sim \frac{\|A_i P\|_2^2}{\|AP\|_F^2}$

Run proposed algorithm on A, then *multiply by P*:

- CountSketch and AMS both are linear transformations

✓ **A** is mapped to **SA**

✓ **S (AP) = (SA) P**

# Handling Post-Processing Matrix

**Input:** matrix **A** as a data stream, a post-processing matrix ***P***

**Output:** index $i$ of a row of **AP** sampled w.p. $\sim \frac{\|A_i P\|_2^2}{\|AP\|_F^2}$

Run proposed algorithm on A, then *multiply by P*:

- CountSketch and AMS both are linear transformations

✓ **A** is mapped to **SA**

✓ **S (AP) = (SA) P**

Total space for sampler: $O(\frac{d}{\epsilon^2}\log^2 n)$ bits

## $L_{2,2}$ sampling with post processing

**Input:**

- $A \in \mathbb{R}^{n \times d}$ as a (turnstile) stream
- a post-processing $\boldsymbol{P} \in \mathbb{R}^{d \times d}$

**Output:** samples an index $i \in [n]$ w.p. $(1 \pm \epsilon)\frac{\|\boldsymbol{A_i P}\|_2^2}{\|\boldsymbol{AP}\|_F^2} + \frac{1}{poly(n)}$

✓ In one pass

✓ $poly(d, \epsilon^{-1}, \log n)$ space

# Adaptive Sampler

1. Simulate adaptive sampling in 1 pass
   - $L_{p,2}$ sampling with post processing matrix $P$
2. Applications in turnstile stream
   - Row/column subset selection
   - Subspace approximation
   - Projective clustering
   - Volume Maximization
3. Volume maximization lower bounds
4. Volume maximization in row arrival

# Algorithm Using $L_{2,2}$ Sampler

Maintain $k$ instances of $L_{2,2}$ sampler with post processing: $S_1, \ldots, S_k$

$M \leftarrow \emptyset$

For round $i = 1$ to $k$,

- Set $P \leftarrow (I - M^+ M)$
- Use $S_i$ to sample a noisy row $r_j$ of $A$ with post processing matrix $P$
- Append $r_j$ to $M$

# Algorithm Using $L_{2,2}$ Sampler

Maintain $k$ instances of $L_{2,2}$ sampler with post processing: $S_1, \ldots, S_k$

$M \leftarrow \emptyset$

For round $i = 1$ to $k$,

- Set $P \leftarrow (I - M^+ M)$
- Use $S_i$ to sample a noisy row $r_j$ of $A$ with post processing matrix $P$
- Append $r_j$ to $M$

**Issues:**

X Noisy perturbation of rows (unavoidable)

- ✓ Sample $j$,
- ✓ $r_j = \mathrm{A}_j P + \mathrm{v}$ where v has small norm $\|\mathrm{v}\| < \epsilon \|A_j P\|$ thus $\|r_j\| \approx \|A_j P\|$

# Algorithm Using $L_{2,2}$ Sampler

Maintain $k$ instances of $L_{2,2}$ sampler with post processing: $\boldsymbol{S_1}, \ldots, \boldsymbol{S_k}$
$M \leftarrow \emptyset$
For round $i = 1$ to $k$,

- Set $P \leftarrow (I - M^+ M)$
- Use $\boldsymbol{S_i}$ to sample a noisy row $r_j$ of $A$ with post processing matrix $P$
- Append $r_j$ to $M$

**Issues:**

X Noisy perturbation of rows (unavoidable)

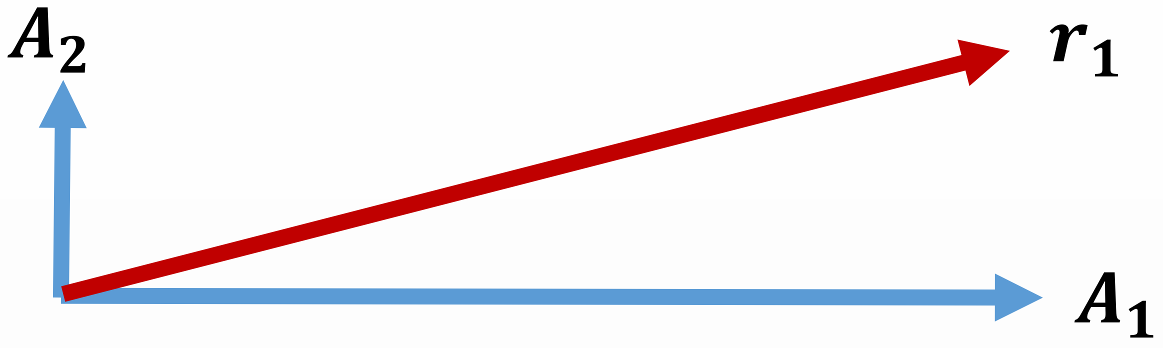
✓ Sample $j$,

✓ $r_j = \mathrm{A}_j P + \mathrm{v}$ where v has small norm $\|\mathrm{v}\| < \epsilon \|A_j P\|$ thus $\|r_j\| \approx \|A_j P\|$

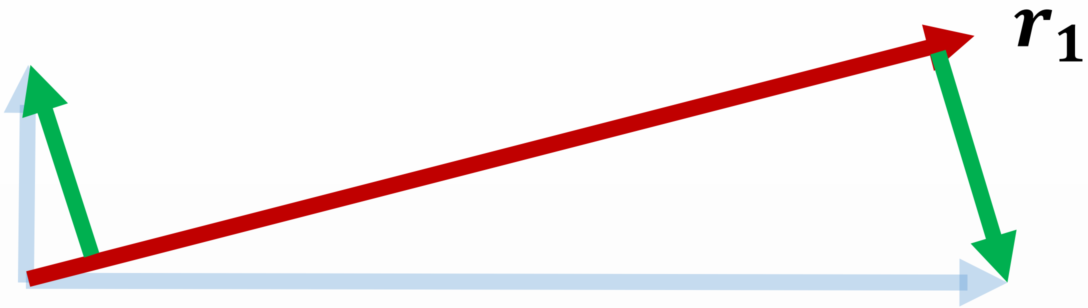
X This can drastically change the probabilities: *may **zero out** probabilities of some rows*

# Bad Example

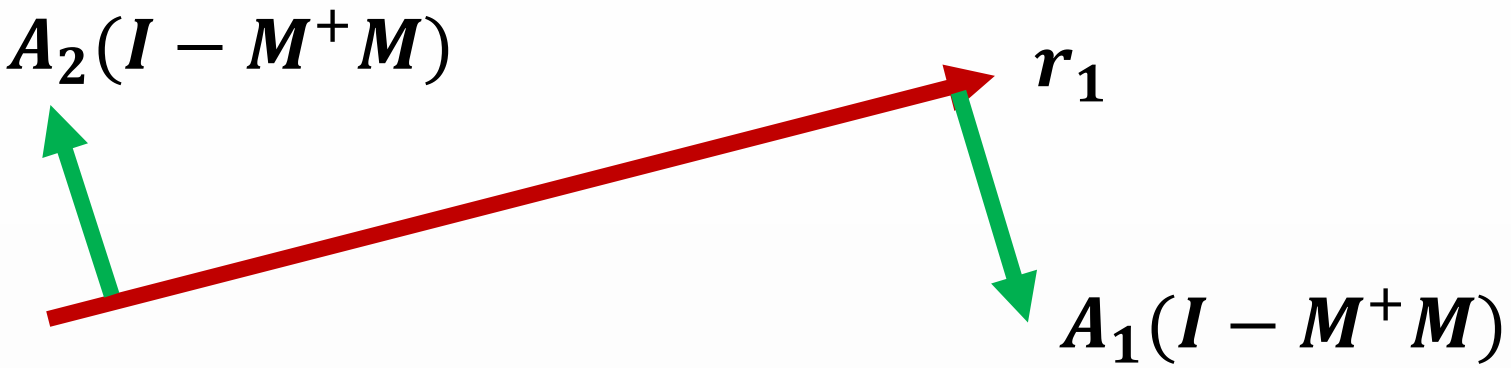
# Bad Example



# Bad Example

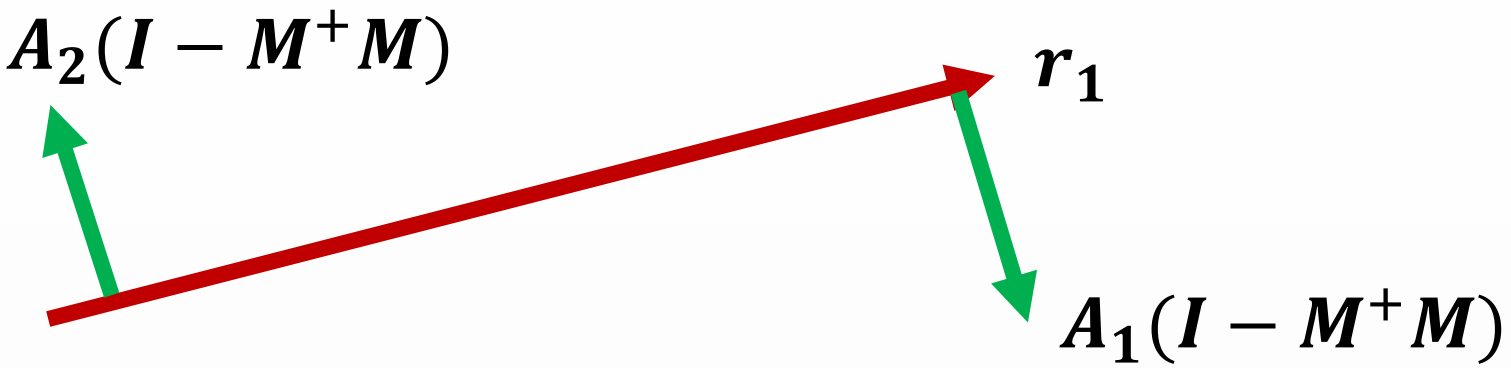


# Bad Example




Noisy row sampling: $\|A_1(I - M^+M)\| \geq \|A_2(I - M^+M)\|$

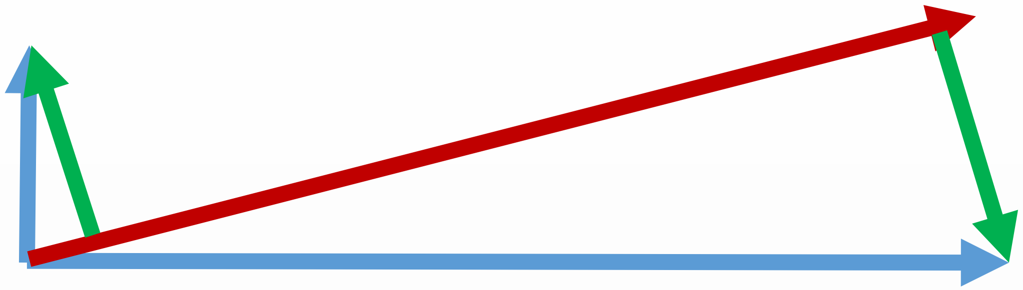
# Bad Example




Noisy row sampling: $\|A_1(I - M^+M)\| \geq \|A_2(I - M^+M)\|$

Sample one row again and again

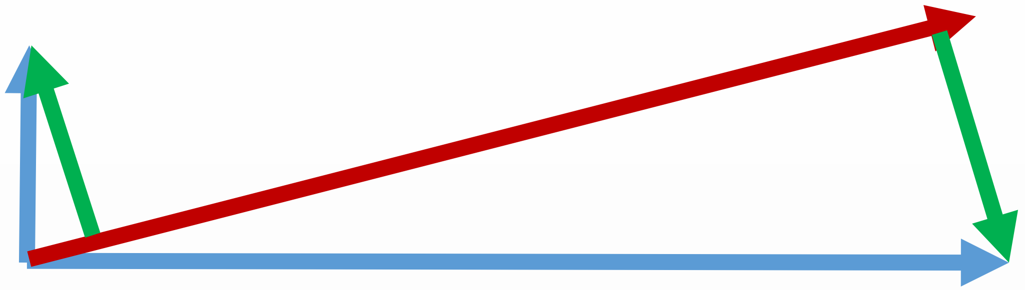
# Bad Example



Noisy row sampling: $\|\boldsymbol{A_1}(\boldsymbol{I}-\boldsymbol{M}^{+}\boldsymbol{M})\| \geq \|\boldsymbol{A_2}(\boldsymbol{I}-\boldsymbol{M}^{+}\boldsymbol{M})\|$

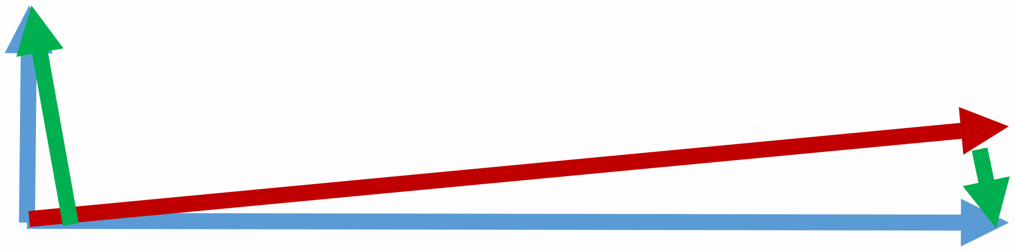
True row sampling: $\|\boldsymbol{A_1}(\boldsymbol{I}-\boldsymbol{M}^{+}\boldsymbol{M})\| = 0$

# Bad Example



- We cannot hope for a multiplicative bound on probabilities.

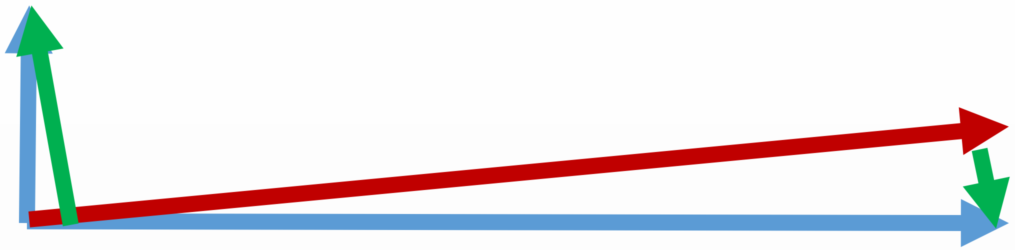
# Bad Example



x We cannot hope for a multiplicative bound on probabilities.

**Lemma:** Not only the norm of $v$ is small in compare to $A_j$ but also its norm projected away from $A_j$ is small

# Bad Example

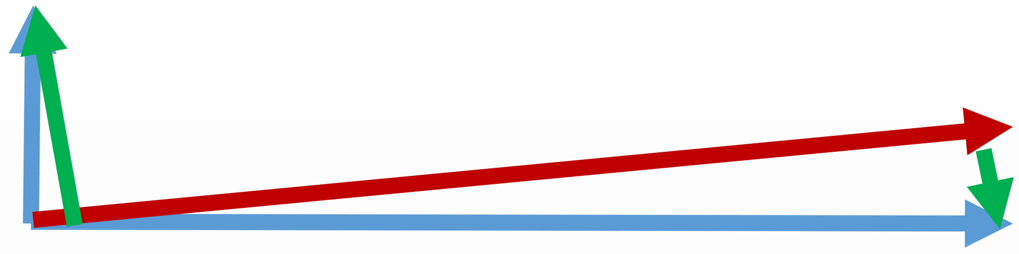


x We cannot hope for a multiplicative bound on probabilities.

**Lemma:** Not only the norm of $v$ is small in compare to $A_j$ but also its norm projected away from $A_j$ is small:

- $r_j = A_j P + v$
- where $\|\boldsymbol{vQ}\| \leq \epsilon \|A_j P\| \cdot \frac{\|\boldsymbol{APQ}\|_F}{\|\boldsymbol{AP}\|_F}$ for **any projection** matrix $\boldsymbol{Q}$

# Bad Example

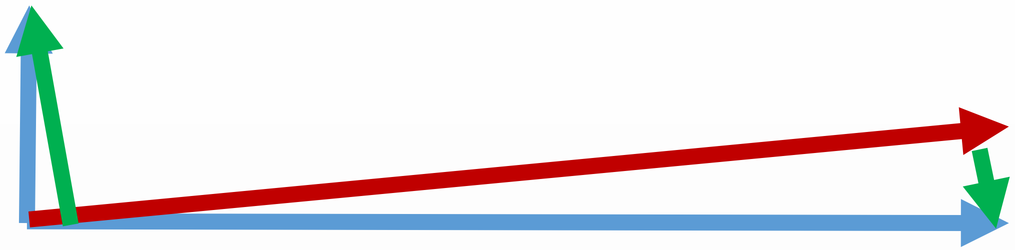


x We cannot hope for a multiplicative bound on probabilities.

**Lemma:** Not only the norm of $v$ is small in compare to $A_j$ but also its norm projected away from $A_j$ is small:

- $r_j = A_j P + v$
- where $\|\boldsymbol{vQ}\| \leq \epsilon \left\| A_j P \right\| \cdot \frac{\|\boldsymbol{APQ}\|_F}{\|\boldsymbol{AP}\|_F}$ for **any projection** matrix $\boldsymbol{Q}$

# Bad Example



x We cannot hope for a multiplicative bound on probabilities.

**Lemma:** Not only the norm of $v$ is small in compare to $A_j$ but also its norm projected away from $A_j$ is small:

- $r_j = A_j P + v$
- where $\|\boldsymbol{v}\boldsymbol{Q}\| \leq \epsilon \|A_j P\| \cdot \frac{\|\boldsymbol{APQ}\|_F}{\|\boldsymbol{AP}\|_F}$ for **any projection** matrix $\boldsymbol{Q}$

✓ **Bound the additive error of sampling probabilities in subsequent rounds**

# Overview of How to Bound the Error

Suppose indices reported by our algorithm are $j_1, \ldots, j_k$

Consider two bases $\boldsymbol{U}$ and $\boldsymbol{W}$

- $\boldsymbol{U}$ follows True rows: $U = \{u_1, \ldots, u_d\}$ s.t. $\{u_1, \ldots, u_i\}$ spans $\{A_{j_1}, \ldots, A_{j_i}\}$
- $\boldsymbol{W}$ follows Noisy rows: $W = \{w_1, \ldots, w_d\}$ s.t. $\{w_1, \ldots, w_i\}$ spans $\{r_{j_1}, \ldots, r_{j_i}\}$

# Overview of How to Bound the Error

Suppose indices reported by our algorithm are $j_1, \dots, j_k$

Consider two bases $\boldsymbol{U}$ and $\boldsymbol{W}$

- $\boldsymbol{U}$ follows True rows: $U = \{u_1, \dots, u_d\}$ s.t. $\{u_1, \dots, u_i\}$ spans $\{A_{j_1}, \dots, A_{j_i}\}$
- $\boldsymbol{W}$ follows Noisy rows: $W = \{w_1, \dots, w_d\}$ s.t. $\{w_1, \dots, w_i\}$ spans $\{r_{j_1}, \dots, r_{j_i}\}$

For row $A_x$:

- $A_x = \sum_{i=1}^{d} \lambda_{x,i} u_i$
- $A_x = \sum_{i=1}^{d} \xi_{x,i} w_i$

# Overview of How to Bound the Error

Suppose indices reported by our algorithm are $j_1, \dots, j_k$

Consider two bases $\boldsymbol{U}$ and $\boldsymbol{W}$

- $\boldsymbol{U}$ follows True rows: $U = \{u_1, \dots, u_d\}$ s.t. $\{u_1, \dots, u_i\}$ spans $\{A_{j_1}, \dots, A_{j_i}\}$
- $\boldsymbol{W}$ follows Noisy rows: $W = \{w_1, \dots, w_d\}$ s.t. $\{w_1, \dots, w_i\}$ spans $\{r_{j_1}, \dots, r_{j_i}\}$

For row $A_x$:

- $A_x = \sum_{i=1}^{d} \lambda_{x,i} u_i$
- $A_x = \sum_{i=1}^{d} \xi_{x,i} w_i$

Sampling probs in terms of $U$ and $W$ in t-th round

- The correct probability: $\dfrac{\sum_{i=t}^{d} \lambda_{x,i}^2}{\sum_{y=1}^{n} \sum_{i=t}^{d} \lambda_{y,i}^2}$
- What we sample from: $\dfrac{\sum_{i=t}^{d} \xi_{x,i}^2}{\sum_{y=1}^{n} \sum_{i=t}^{d} \xi_{y,i}^2}$

# Overview of How to Bound the Error

Suppose indices reported by our algorithm are $j_1, \dots, j_k$

Consider two bases $\boldsymbol{U}$ and $\boldsymbol{W}$

- $\boldsymbol{U}$ follows True rows: $U = \{u_1, \dots, u_d\}$ s.t. $\{u_1, \dots, u_i\}$ spans $\{A_{j_1}, \dots, A_{j_i}\}$
- $\boldsymbol{W}$ follows Noisy rows: $W = \{w_1, \dots, w_d\}$ s.t. $\{w_1, \dots, w_i\}$ spans $\{r_{j_1}, \dots, r_{j_i}\}$

For row $A_x$:

- $A_x = \sum_{i=1}^{d} \lambda_{x,i} u_i$
- $A_x = \sum_{i=1}^{d} \xi_{x,i} w_i$

Sampling probs in terms of $U$ and $W$ in t-th round

- The correct probability: $\dfrac{\sum_{i=t}^{d} \lambda_{x,i}^2}{\sum_{y=1}^{n} \sum_{i=t}^{d} \lambda_{y,i}^2}$
- What we sample from: $\dfrac{\sum_{i=t}^{d} \xi_{x,i}^2}{\sum_{y=1}^{n} \sum_{i=t}^{d} \xi_{y,i}^2}$

- Difference between the *correct prob* and *our algorithm sampling prob* over all rows is $\epsilon$ for one round
  - Change of basis matrix ≈ Identity matrix
  - Bound total variation distance by $\epsilon$
- Error in each round gets propagated $k$ times
- Total error is $O(k^2 \epsilon)$

**Theorem:**

Our algorithm reports a set of $k$ indices such that with high probability

- the total variation distance between the probability distribution output by the algorithm and the probability distribution of adaptive sampling is at most $O(\epsilon)$
- The algorithm uses space $poly(k, \frac{1}{\epsilon}, d, \log n)$

# Applications

1. Simulate adaptive sampling in 1 pass
   - $L_{p,2}$ sampling with post processing matrix $P$
2. Applications in turnstile stream
   - Row/column subset selection
   - Subspace approximation
   - Projective clustering
   - Volume Maximization
3. Volume maximization lower bounds
4. Volume maximization in row arrival

# Applications

**Main Challenge:** it suffices to get a noisy perturbation of the rows

# Applications: Row Subset Selection

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$ rows of $A$ to form $\boldsymbol{M}$ to minimize $||A - AM^{+}M||_F$

# Applications: Row Subset Selection

**Adaptive Sampling provides a $(k+1)!$ approximation for subset selection**

- **[DRVW'06]**: Volume Sampling provides a $(k+1)$ factor approximation to row subset selection with constant probability.
- **[DV'06]:** Sampling probabilities for any $k$-set $S$ produced by Adaptive Sampling is at most $k!$ of its sampling probability with respect to volume sampling.

# Applications: Row Subset Selection

**Adaptive Sampling provides a $(\boldsymbol{k}+\mathbf{1})!$ approximation for subset selection**

- **[DRVW'06]**: Volume Sampling provides a $(k+1)$ factor approximation to row subset selection with constant probability.
- **[DV'06]:** Sampling probabilities for any $k$-set $S$ produced by Adaptive Sampling is at most $k!$ of its sampling probability with respect to volume sampling.

**Non-adaptive Adaptive Sampling provides a good approximation to Adaptive Sampling**

# Applications: Row Subset Selection

**Adaptive Sampling provides a $(k+1)!$ approximation for subset selection**

- **[DRVW'06]**: Volume Sampling provides a $(k+1)$ factor approximation to row subset selection with constant probability.
- **[DV'06]:** Sampling probabilities for any $k$-set $S$ produced by Adaptive Sampling is at most $k!$ of its sampling probability with respect to volume sampling.

**Non-adaptive Adaptive Sampling provides a good approximation to Adaptive Sampling**

1. For a set of indices $\boldsymbol{J}$ output by our algorithm, $\|A(I - R^+R)\|_F \leq (1+\epsilon)\|A(I - M^+M)\|_F$, w.h.p.
   - $R$: the set of noisy rows corresponding to $\boldsymbol{J}$
   - $M$: the set of true rows corresponding to $\boldsymbol{J}$

# Applications: Row Subset Selection

**Adaptive Sampling provides a $(k+1)!$ approximation for subset selection**

- **[DRVW'06]**: Volume Sampling provides a $(k+1)$ factor approximation to row subset selection with constant probability.
- **[DV'06]:** Sampling probabilities for any $k$-set $S$ produced by Adaptive Sampling is at most $k!$ of its sampling probability with respect to volume sampling.

**Non-adaptive Adaptive Sampling provides a good approximation to Adaptive Sampling**

1. For a set of indices $\boldsymbol{J}$ output by our algorithm, $\|A(I-R^{+}R)\|_{\mathrm{F}} \leq (1+\epsilon)\|A(I-M^{+}M)\|_{\mathrm{F}}$, w.h.p.
   - $R$: the set of noisy rows corresponding to $\boldsymbol{J}$
   - $M$: the set of true rows corresponding to $\boldsymbol{J}$
2. For most $k$-sets $\boldsymbol{J}$, its prob. by adaptive sampling is within $O(1)$ factor of Non-adaptive Sampling.

# Applications: Row Subset Selection

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k} > 0$

**Output:** $\boldsymbol{k}$ rows of $A$ to form $\boldsymbol{M}$ to minimize $\|A - AM^{+}M\|_F$

**Our Result:** finds **M** such that,

$$\mathbf{Pr}[\|A - A\boldsymbol{M}^{+}\boldsymbol{M}\|_F^2 \leq 16(k+1)!\,\|A - A_k\|_F^2] \geq 2/3$$

- $A_k$: best rank-*k* approximation of $A$
- first one pass turnstile streaming algorithm
- $poly(d, k, \log n)$ space

# Applications: Volume Maximization

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

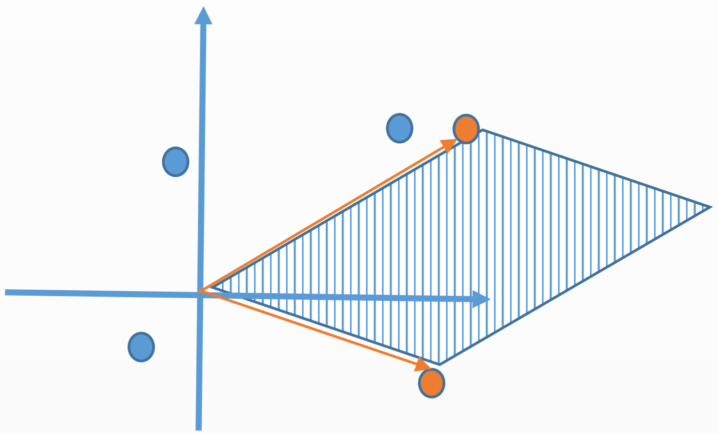
# Applications: Volume Maximization

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$

**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \ldots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

Volume of the parallelepiped spanned by those vectors

$\boldsymbol{k = 2}$

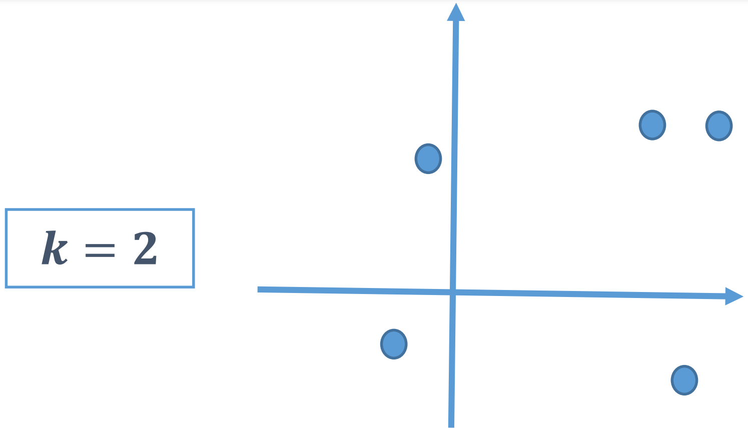


# Applications: Volume Maximization

[Civril, Magdon'09] **Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Greedy**

- For $k$ rounds, pick the vector that is farthest away from the current subspace.



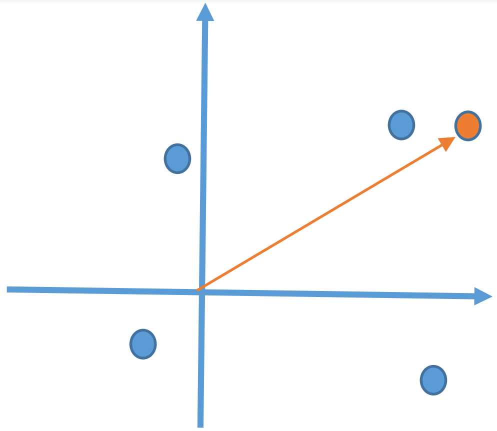
# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Greedy**

- For $k$ rounds, pick the vector that is farthest away from the current subspace.

$k = 2$



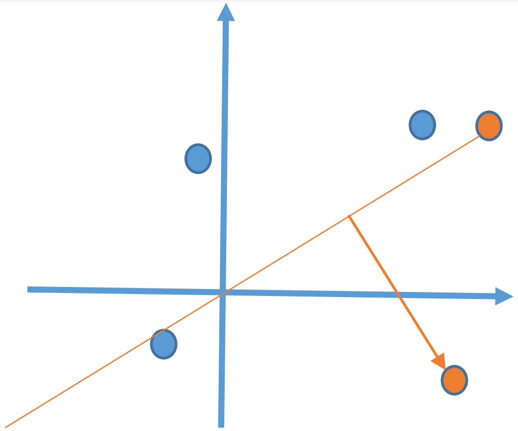
# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Greedy**

- For $k$ rounds, pick the vector that is farthest away from the current subspace.

$k = 2$



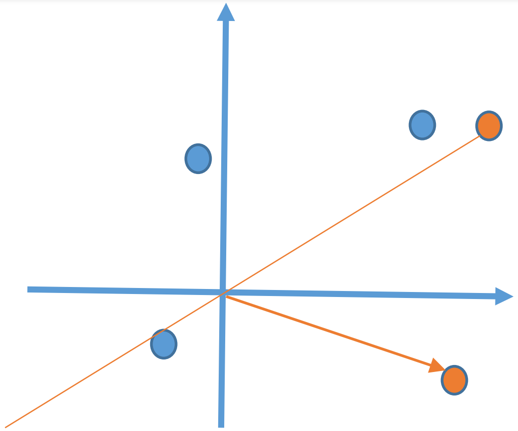
# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Greedy**

- For $k$ rounds, pick the vector that is farthest away from the current subspace.

$k = 2$

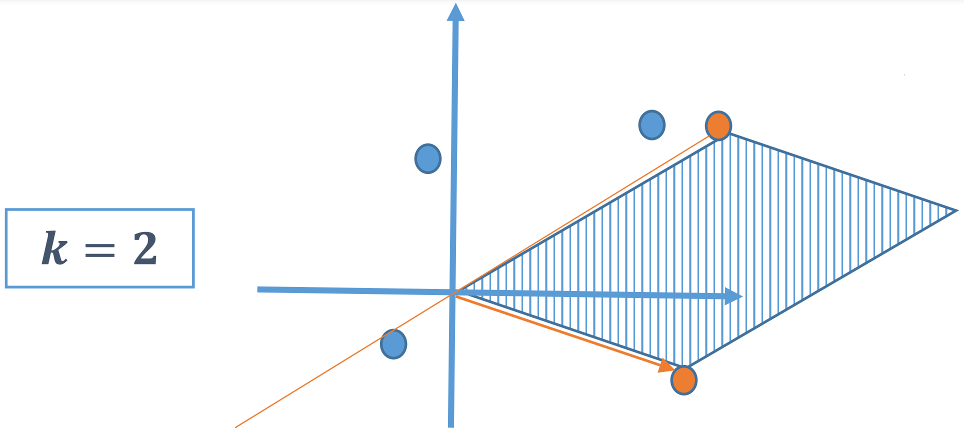


# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Greedy**

- For $k$ rounds, pick the vector that is farthest away from the current subspace.



# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Simulate Greedy**

- Maintain $k$ instances of **CountSketch**, **AMS** and $L_{2,2}$ **Sampler**

# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Simulate Greedy**

- Maintain $k$ instances of **CountSketch**, **AMS** and $L_{2,2}$ **Sampler**
- For $k$ rounds,
  - Let $\boldsymbol{r}$ be the row of $AP$ with largest norm //by **CountSketch**

➢ If the largest row exceeds the threshold, then it is correctly found by CountSketch w.h.p.

# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Simulate Greedy**

- Maintain $k$ instances of **CountSketch**, **AMS** and $L_{2,2}$ **Sampler**
- For $k$ rounds,
  - Let $r$ be the row of $AP$ with largest norm //by **CountSketch**
  - If $\|r\|^2 < \frac{\alpha^2}{4nk}\|AP\|_F^2$, instead **sample row** $r$ according to norms of rows

➢ If the largest row exceeds the threshold, then it is correctly found by CountSketch w.h.p.

➢ Otherwise, there are enough large rows and sampler chooses one of them w.h.p.

# Applications: Volume Maximization

**[Civril, Magdon'09] Greedy Algorithm Provides a $k!$ approximation to Volume Maximization**

**Simulate Greedy**

- Maintain $k$ instances of **CountSketch**, **AMS** and $\boldsymbol{L_{2,2}}$ **Sampler**
- For $k$ rounds,
  - Let $\boldsymbol{r}$ be the row of $AP$ with largest norm //by **CountSketch**
  - If $\|\boldsymbol{r}\|^2 < \frac{\alpha^2}{4nk}\|\boldsymbol{AP}\|_F^2$, instead **sample row $\boldsymbol{r}$** according to norms of rows
  - Add $r$ to the solution, and update the postprocessing matrix $P$

➢ If the largest row exceeds the threshold, then it is correctly found by CountSketch w.h.p.

➢ Otherwise, there are enough large rows and sampler chooses one of them w.h.p.

# Applications: Volume Maximization

**Input:** $A \in \mathbb{R}^{n \times d}$ and an integer $\boldsymbol{k}$
**Output:** $\boldsymbol{k}$ rows $\boldsymbol{r_1}, \dots, \boldsymbol{r_k}$ of $A$, $\boldsymbol{M}$, with maximum volume

**Our Result:** for an approximation factor $\boldsymbol{\alpha}$, finds **S** (set of ***k* noisy rows of *A***) s.t.,

$$\mathbf{Pr}[\alpha^k (k!) \mathrm{Vol}(\mathbf{S}) \geq \mathrm{Vol}(\mathbf{M})] \geq 2/3$$

- first one pass turnstile streaming algorithm
- $\tilde{O}(ndk^2/\alpha^2)$ space

| Problem | Model | Approximation/error | space | Comments |
|---|---|---|---|---|
| $L_{p,2}$ Sampler | turnstile | $(1+\epsilon)$ relative + $\frac{1}{poly(n)}$ | $poly(d, \epsilon^{-1}, \log n)$ | |
| Adaptive Sampling | | $O(\epsilon)$ total variation distance | $poly(d, k, \epsilon^{-1}, \log n)$ | |
| Row Subset Selection | | $O((k+1)!)$ | $poly(d, k, \log n)$ | |
| Subspace Approximation | | $O((k+1)!)$ | $poly(d, k, \log n)$ | |
| | | $(1+\epsilon)$ | $poly(d, k, \log n, 1/\epsilon)$ | $poly(k, 1/\epsilon)$ rows |
| Projective Clustering | | $(1+\epsilon)$ | $poly(d, k, \log n, s, 1/\epsilon)$ | $poly(k, s, 1/\epsilon)$ rows |
| Volume Maximization | | $\alpha^k(k!)$ | $\tilde{O}(ndk^2/\alpha^2)$ | |
| | | $\alpha^k$ | $\Omega(n/kp\alpha^2)$ | $p$ pass |
| | Row Arrival | $C^k$ | $\Omega(n)$ | Random Order |
| | | $\tilde{O}(Ck)^{k/2}$ | $\tilde{O}(n^{O(1/C)}d)$ | $C < (\log n)/k$ |

| Problem | Model | Approximation/error | space | Comments |
|---|---|---|---|---|
| $L_{p,2}$ Sampler | turnstile | $(1+\epsilon)$ relative + $\frac{1}{poly(n)}$ | $poly(d, \epsilon^{-1}, \log n)$ | |
| Adaptive Sampling | | $O(\epsilon)$ total variation distance | $poly(d, k, \epsilon^{-1}, \log n)$ | |
| Row Subset Selection | | $O((k+1)!)$ | $poly(d, k, \log n)$ | |
| Subspace Approximation | | $O((k+1)!)$ | $poly(d, k, \log n)$ | |
| | | $(1+\epsilon)$ | $poly(d, k, \log n, 1/\epsilon)$ | $poly(k, 1/\epsilon)$ rows |
| Projective Clustering | | $(1+\epsilon)$ | $poly(d, k, \log n, s, 1/\epsilon)$ | $poly(k, s, 1/\epsilon)$ rows |
| Volume Maximization | | $\alpha^k(k!)$ | $\tilde{O}(ndk^2/\alpha^2)$ | |
| | | $\alpha^k$ | $\Omega(n/kp\alpha^2)$ | $p$ pass |
| | Row Arrival | $C^k$ | $\Omega(n)$ | Random Order |
| | | $\tilde{O}(Ck)^{k/2}$ | $\tilde{O}(n^{O(1/C)}d)$ | $C < (\log n)/k$ |

## Open problems

- Get tight dependence on the parameters
- Further applications of non-adaptive adaptive sampling
- Result on Volume Maximization in row arrival model is not tight, i.e., can we get $O(k)^k$ approximation without dependence on $n$?

| Problem | Model | Approximation/error | space | Comments |
|---|---|---|---|---|
| $L_{p,2}$ Sampler | turnstile | $(1+\epsilon)$ relative + $\frac{1}{poly(n)}$ | $poly(d, \epsilon^{-1}, \log n)$ | |
| Adaptive Sampling | | $O(\epsilon)$ total variation distance | $poly(d, k, \epsilon^{-1}, \log n)$ | |
| Row Subset Selection | | $O((k+1)!)$ | $poly(d, k, \log n)$ | |
| Subspace Approximation | | $O((k+1)!)$ | $poly(d, k, \log n)$ | |
| | | $(1+\epsilon)$ | $poly(d, k, \log n, 1/\epsilon)$ | $poly(k, 1/\epsilon)$ rows |
| Projective Clustering | | $(1+\epsilon)$ | $poly(d, k, \log n, s, 1/\epsilon)$ | $poly(k, s, 1/\epsilon)$ rows |
| Volume Maximization | | $\alpha^k(k!)$ | $\tilde{O}(ndk^2/\alpha^2)$ | |
| | | $\alpha^k$ | $\Omega(n/kp\alpha^2)$ | $p$ pass |
| | Row Arrival | $C^k$ | $\Omega(n)$ | Random Order |
| | | $\tilde{O}(Ck)^{k/2}$ | $\tilde{O}(n^{O(1/C)}d)$ | $C < (\log n)/k$ |

## Open problems

- Get tight dependence on the parameters
- Further applications of non-adaptive adaptive sampling
- Result on Volume Maximization in row arrival model is not tight, i.e., can we get $O(k)^k$ approximation without dependence on $n$?

# THANK YOU!