RIDGE LEVERAGE SCORES FOR LOW-RANK MATRIX APPROXIMATION

Michael B. Cohen, Cameron Musco, Christopher Musco

Massachusetts Institute of Technology

"Ridge Leverage Scores for Low-Approximation" =

"Dimensionality Reduction for k-Means Clustering and Low-Rank Approximation"

+

"Uniform Sampling for Matrix Approximation"

“Ridge Leverage Scores for Low-Approximation” =

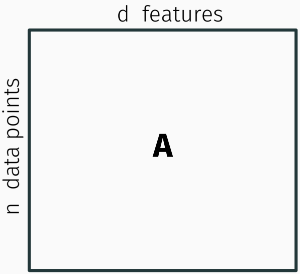
“Dimensionality Reduction for k-Means Clustering and Low-Rank Approximation”

+

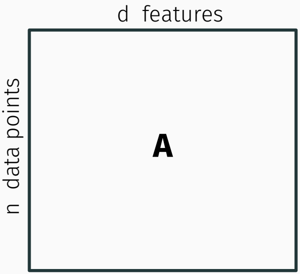
“Uniform Sampling for Matrix Approximation”

Papers and slides available at `chrismusco.com`.

HOW TO DEAL WITH HUGE DATA SETS?

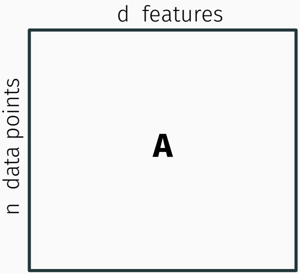


HOW TO DEAL WITH HUGE DATA SETS?



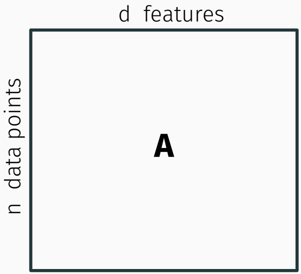
- computing power (MapReduce/Hadoop, Apache Spark, etc.)

HOW TO DEAL WITH HUGE DATA SETS?



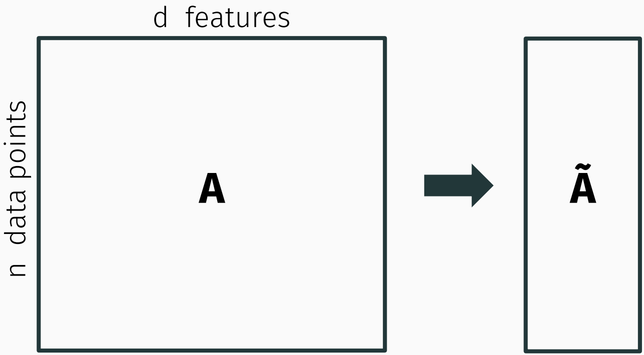
- computing power (MapReduce/Hadoop, Apache Spark, etc.)
- limited data access (iterative methods, stochastic methods)

HOW TO DEAL WITH HUGE DATA SETS?

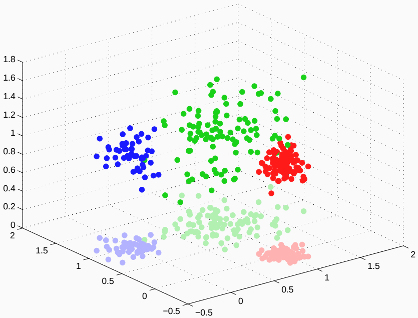


- computing power (MapReduce/Hadoop, Apache Spark, etc.)
- limited data access (iterative methods, stochastic methods)
- dimensionality reduction (“sketch-and-solve”)

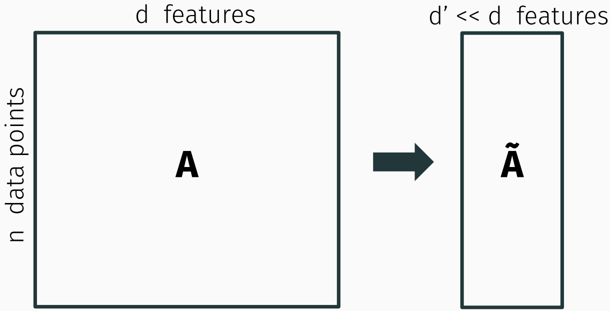
Replace high dimensional data with low dimensional sketch.



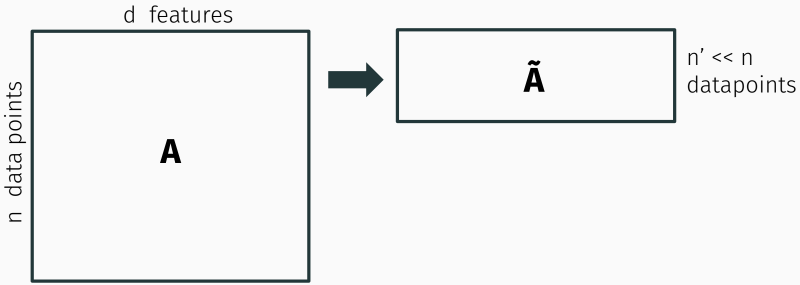
Solution on sketch $\tilde{\mathbf{A}}$ should approximate original solution.



Replace dimensional of data points, not their number.



Reduce the number of data points, not their dimension.



Ã is often called a coreset.

There are tons of sketching techniques, each with their own advantages and disadvantages.

METHODS OVERVIEW

There are tons of sketching techniques, each with their own advantages and disadvantages.

- **Johnson-Lindenstrauss projections** = super fast to apply, naturally adapts to streaming/distributed environments.

There are tons of sketching techniques, each with their own advantages and disadvantages.

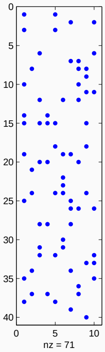
- **Johnson-Lindenstrauss projections** = super fast to apply, naturally adapts to streaming/distributed environments.
- **Deterministic methods (SVD, Frequent Directions)** = best data compression.

There are tons of sketching techniques, each with their own advantages and disadvantages.

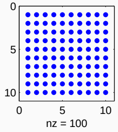
- **Johnson-Lindenstrauss projections** = super fast to apply, naturally adapts to streaming/distributed environments.
- **Deterministic methods (SVD, Frequent Directions)** = best data compression.
- **Data Selection/Sampling** = preserves structure and sparsity.

SKETCHING BY SAMPLING

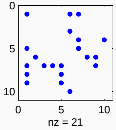
Original Data



General Sketch

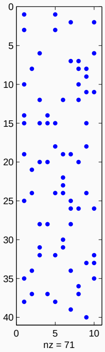


Data Sample

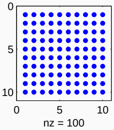


SKETCHING BY SAMPLING

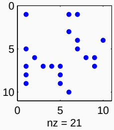
Original Data



General Sketch

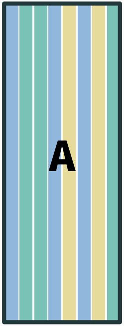


Data Sample



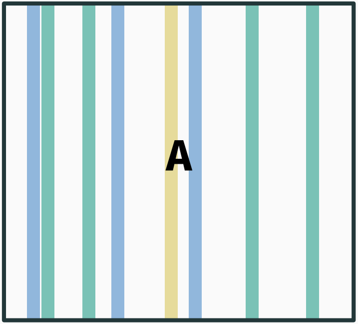
Sampling is also closely tied to understanding heuristic methods and has produced valuable theory.

Uniformly sampling data rarely works (imagine adding a bunch of all-zeros columns to $\mathbf{A}$).



Sketching by sampling is all about understanding which sampling probability to assign to each column in $\mathbf{A}$.

Uniformly sampling data rarely works (imagine adding a bunch of all-zeros columns to $\mathbf{A}$).



Sketching by sampling is all about understanding which sampling probability to assign to each column in $\mathbf{A}$.

WHAT'S THIS PAPER ABOUT?

1. Leverage Scores are used ubiquitously as importance sampling probabilities for matrix sketching.

1. Leverage Scores are used ubiquitously as importance sampling probabilities for matrix sketching.
2. These scores have been extended to sketches for low-rank approximation problems, but not in a satisfying way.

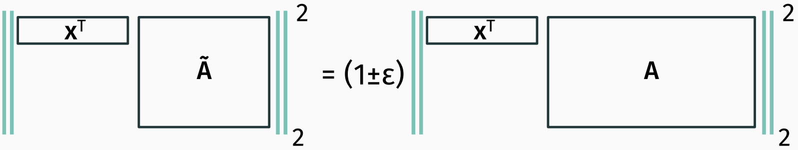
1. Leverage Scores are used ubiquitously as importance sampling probabilities for matrix sketching.
2. These scores have been extended to sketches for low-rank approximation problems, but not in a satisfying way.
3. We give a more natural extension, via Ridge Leverage Scores. These scores lead to simple proofs and have a bunch of desirable properties and new applications.

Definition (Subspace Embedding)

A sketch $\tilde{\mathbf{A}}$ such that, for all vectors $\mathbf{x}$, $\|\mathbf{x}^T\tilde{\mathbf{A}}\| = (1 \pm \epsilon)\|\mathbf{x}^T\mathbf{A}\|$.

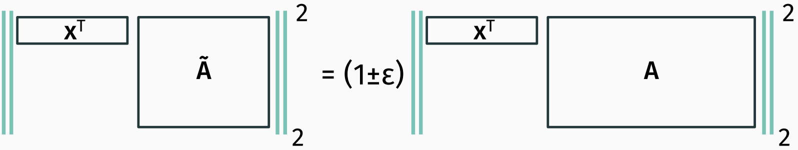
Definition (Subspace Embedding)

A sketch $\tilde{A}$ such that, for all vectors x, $\|x^T\tilde{A}\| = (1 \pm \epsilon)\|x^T A\|$.



Definition (Subspace Embedding)

A sketch $\tilde{A}$ such that, for all vectors x, $\|x^T\tilde{A}\| = (1 \pm \epsilon)\|x^T A\|$.

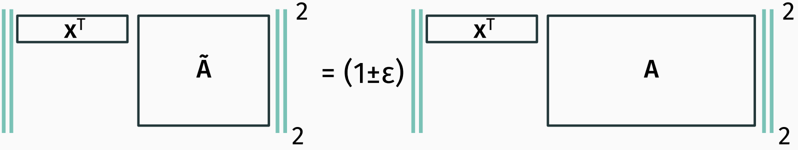


Applications:

- Approximate (constrained) linear regression.

Definition (Subspace Embedding)

A sketch $\tilde{\mathbf{A}}$ such that, for all vectors $\mathbf{x}$, $\|\mathbf{x}^T\tilde{\mathbf{A}}\| = (1 \pm \epsilon)\|\mathbf{x}^T\mathbf{A}\|$.

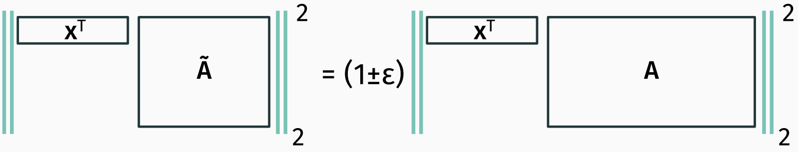


Applications:

- Approximate (constrained) linear regression.
- Constructing preconditioners for iterative system solvers.

Definition (Subspace Embedding)

A sketch $\tilde{\mathbf{A}}$ such that, for all vectors $\mathbf{x}$, $\|\mathbf{x}^T\tilde{\mathbf{A}}\| = (1 \pm \epsilon)\|\mathbf{x}^T\mathbf{A}\|$.



Applications:

- Approximate (constrained) linear regression.
- Constructing preconditioners for iterative system solvers.
- Spectral sparsifiers for fast approximate graph algorithms.

Equivalent formulation of subspace embeddings:

$$\|\mathbf{x}^T\mathbf{A}\|_2^2 = (1 \pm \epsilon)\|\mathbf{x}^T\tilde{\mathbf{A}}\|_2^2$$

Equivalent formulation of subspace embeddings:

$$\mathbf{x}^T \mathbf{A}\mathbf{A}^T \mathbf{x} = (1 \pm \epsilon)\mathbf{x}^T \tilde{\mathbf{A}}\tilde{\mathbf{A}}^T \mathbf{x}$$

Equivalent formulation of subspace embeddings:

$$\mathbf{x}^T \mathbf{A}\mathbf{A}^T \mathbf{x} = (1 \pm \epsilon) \mathbf{x}^T \tilde{\mathbf{A}}\tilde{\mathbf{A}}^T \mathbf{x}$$

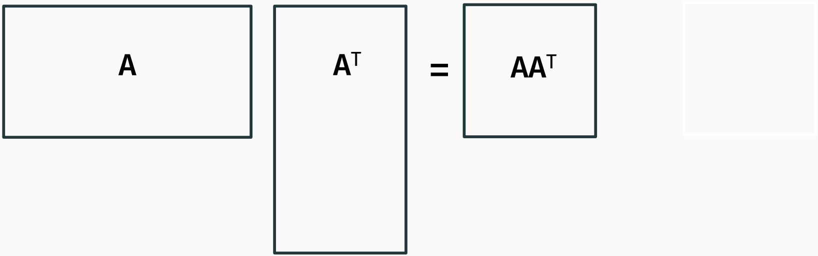
$$(1 - \epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T \preceq \mathbf{A}\mathbf{A}^T \preceq (1 + \epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T$$

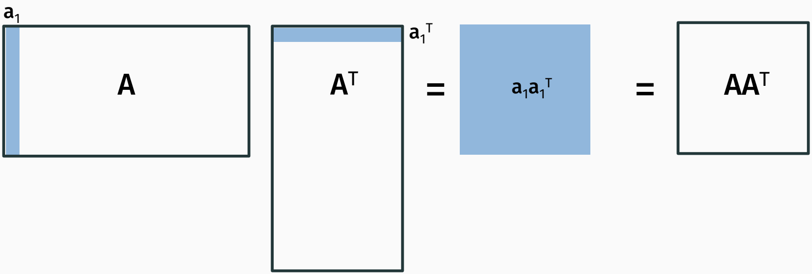
Equivalent formulation of subspace embeddings:

$$\mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x} = (1 \pm \epsilon)\mathbf{x}^T\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\mathbf{x}$$

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T$$

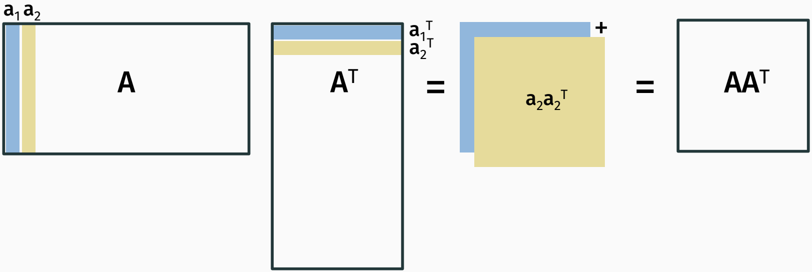
Let's think about subspace embeddings as approximating the quadratic form $\mathbf{A}\mathbf{A}^T$.

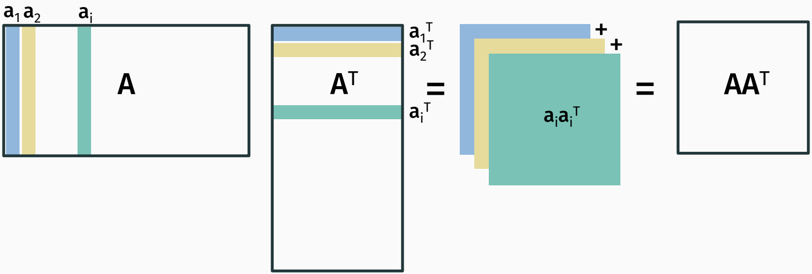

A
A^T
=
AA^T

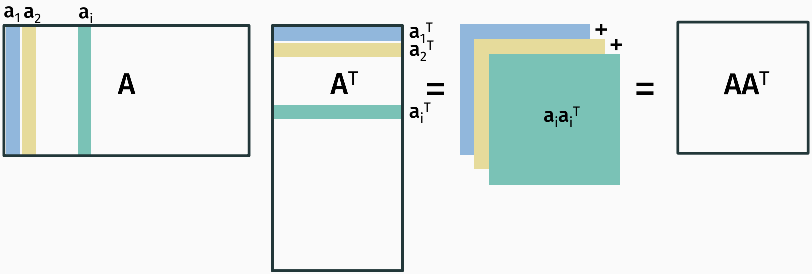


a₁
A
Aᵀ
a₁ᵀ
=
a₁a₁ᵀ
=
AAᵀ

QUADRATIC FORM SAMPLING




a1 a2 ai
A
AT
a1T
a2T
aiT
=
+
+
aiaiT
=
AAT



$$AA^T = \sum_{i=1}^{d} a_i a_i^T$$

Sampling Scheme: For any set of sampling probabilities $p_1, p_2, \ldots, p_d$ include column $\mathbf{a}_i$ in $\tilde{\mathbf{A}}$ with probability p_i and reweight the column by $\frac{1}{p_i}$.

Sampling Scheme: For any set of sampling probabilities $p_1, p_2, \ldots, p_d$ include column $\mathbf{a}_i$ in $\tilde{\mathbf{A}}$ with probability p_i and reweight the column by $\frac{1}{p_i}$.

Then:

$$\mathbb{E}\left[\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\right] = \sum_{i=1}^{d} p_i \cdot \left(\frac{1}{p_i}\mathbf{a}_i\mathbf{a}_i^T\right)$$

Sampling Scheme: For any set of sampling probabilities $p_1, p_2, \ldots, p_d$ include column $\mathbf{a}_i$ in $\tilde{\mathbf{A}}$ with probability p_i and reweight the column by $\frac{1}{p_i}$.

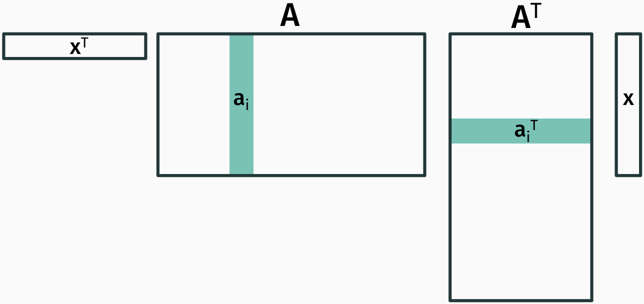
Then:

$$\mathbb{E}\left[\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\right] = \sum_{i=1}^{d} p_i \cdot \left(\frac{1}{p_i}\mathbf{a}_i\mathbf{a}_i^T\right) = \sum_{i=1}^{d} \mathbf{a}_i\mathbf{a}_i^T = \mathbf{A}\mathbf{A}^T$$

How to get good concentration?

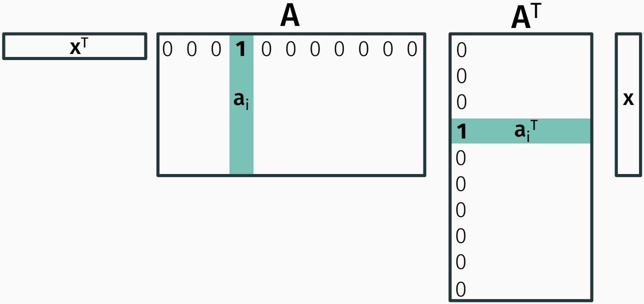
How to get good concentration?

Need to select more "unique" columns with higher probability.



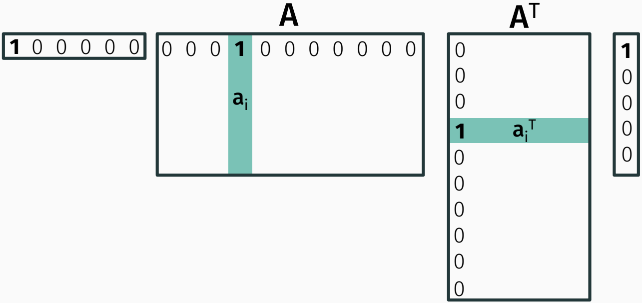
How to get good concentration?

Need to select more "unique" columns with higher probability.



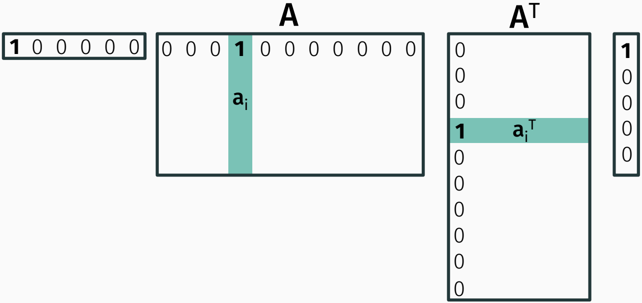
How to get good concentration?

Need to select more "unique" columns with higher probability.



How to get good concentration?

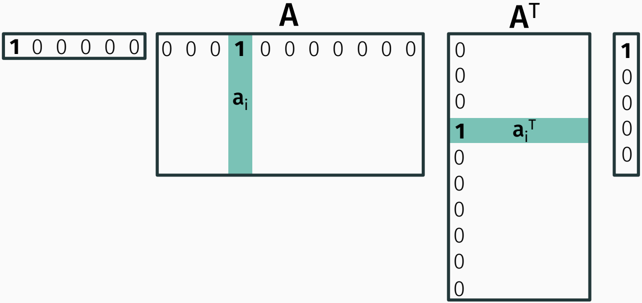
Need to select more "unique" columns with higher probability.



If we don't select $\mathbf{a}_i$ then $\mathbf{x}^T\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\mathbf{x} = 0$, while $\mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x}$ is positive.

How to get good concentration?

Need to select more "unique" columns with higher probability.



If we don't select $\mathbf{a}_i$ then $\mathbf{x}^T\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\mathbf{x} = 0$, while $\mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x}$ is positive.

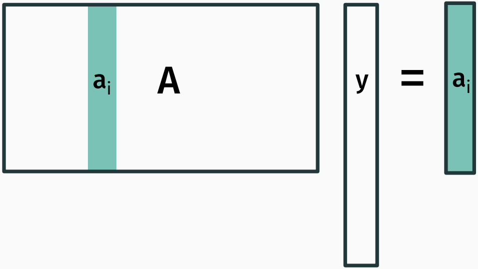
$\mathbf{x}^T\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\mathbf{x}$ cannot equal $(1 \pm \epsilon)\mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x}$.

How to measure “unique-ness”:

How to measure "unique-ness":

Definition (Leverage Score, τ)

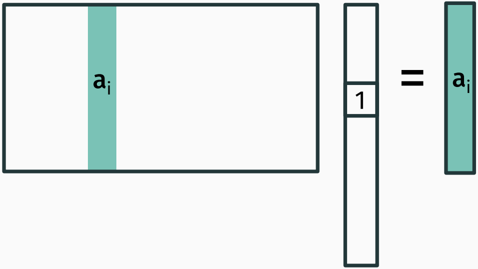
$\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$



How to measure “unique-ness”:

Definition (Leverage Score, τ)

$\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$

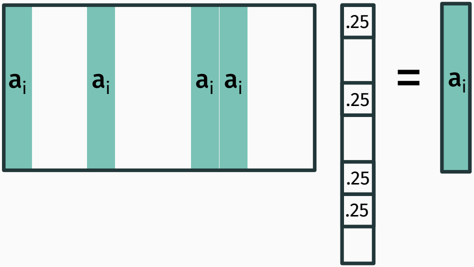


$\tau(\mathbf{a}_i) \leq 1$ since we can choose $\mathbf{y}$ to be the i^{th} basis vector.

How to measure "unique-ness":

Definition (Leverage Score, τ)

$\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$



If more columns align with $\mathbf{a}_i$, $\tau(\mathbf{a}_i)$ decreases.

Problem: Find $\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$.

Solution:

Problem: Find $\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$.

Solution:

$$\mathbf{y} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{a}_i$$

Problem: Find $\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$.

Solution:

$$\mathbf{y} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{a}_i$$

$$\tau(\mathbf{a}_i) = \|\mathbf{y}\|_2^2 = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$$

Problem: Find $\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$.

Solution:

$$\mathbf{y} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{a}_i$$

$$\tau(\mathbf{a}_i) = \|\mathbf{y}\|_2^2 = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$$

$\sum_i \tau(\mathbf{a}_i) = \text{tr}(\mathbf{A}^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A})$.

Problem: Find $\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2$ such that $\mathbf{a}_i = \mathbf{A}\mathbf{y}$.

Solution:

$$\mathbf{y} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{a}_i$$

$$\tau(\mathbf{a}_i) = \|\mathbf{y}\|_2^2 = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$$

$\sum_i \tau(\mathbf{a}_i) = \mathrm{tr}(\mathbf{A}^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}) = \mathrm{rank}(\mathbf{A}) \leq n.$

More specifically, to get a subspace embedding, we sample each column $\mathbf{a}_i$ with probability $\tau(\mathbf{a}_i) \cdot \frac{\log n}{\epsilon^2}$.

More specifically, to get a subspace embedding, we sample each column $\mathbf{a}_i$ with probability $\tau(\mathbf{a}_i) \cdot \frac{\log n}{\epsilon^2}$.

We're approximating $\mathbf{A}$ with a sum of (binary) random matrices:

$$\mathbf{X}_i = \begin{cases} \frac{1}{p_i}\mathbf{a}_i\mathbf{a}_i^T & \text{with probability } p_i \\ \mathbf{0} & \text{with probability } (1 - p_i) \end{cases}$$

$\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T = \sum_{i=1}^{d} \mathbf{X}_i$.

More specifically, to get a subspace embedding, we sample each column $\mathbf{a}_i$ with probability $\tau(\mathbf{a}_i) \cdot \frac{\log n}{\epsilon^2}$.

We're approximating $\mathbf{A}$ with a sum of (binary) random matrices:

$$\mathbf{X}_i = \begin{cases} \frac{1}{p_i}\mathbf{a}_i\mathbf{a}_i^T & \text{with probability } p_i \\ \mathbf{0} & \text{with probability } (1-p_i) \end{cases}$$

$\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T = \sum_{i=1}^{d} \mathbf{X}_i$.

$\tau(\mathbf{a}_i)\frac{\log n}{\epsilon^2}$ is the lowest p_i which ensures $\frac{1}{p_i}\mathbf{a}_i\mathbf{a}_i^T \preceq \frac{\epsilon^2}{\log n}\mathbf{A}\mathbf{A}^T$.

More specifically, to get a subspace embedding, we sample each column $\mathbf{a}_i$ with probability $\tau(\mathbf{a}_i) \cdot \frac{\log n}{\epsilon^2}$.

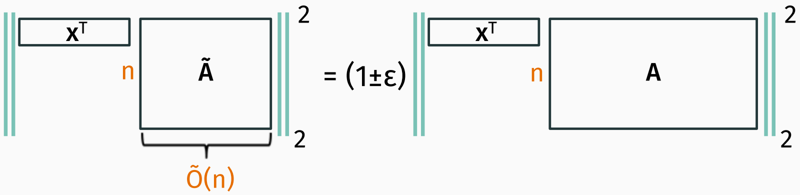
We're approximating $\mathbf{A}$ with a sum of (binary) random matrices:

$$\mathbf{X}_i = \begin{cases} \frac{1}{p_i}\mathbf{a}_i\mathbf{a}_i^T & \text{with probability } p_i \\ \mathbf{0} & \text{with probability } (1 - p_i) \end{cases}$$

$\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T = \sum_{i=1}^{d} \mathbf{X}_i$.

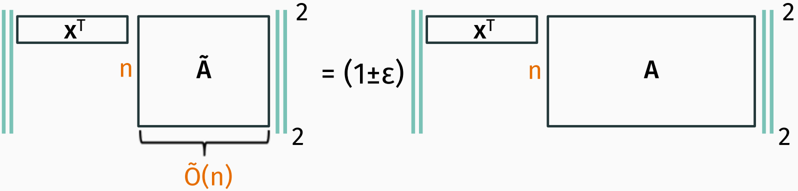
$\tau(\mathbf{a}_i)\frac{\log n}{\epsilon^2}$ is the lowest p_i which ensures $\frac{1}{p_i}\mathbf{a}_i\mathbf{a}_i^T \preceq \frac{\epsilon^2}{\log n}\mathbf{A}\mathbf{A}^T$.

"User-friendly tail bounds for sums of random matrices", Joel Tropp



Theorem (Subspace Embedding via Sampling)

Sampling $O\left(\frac{n \log n}{\epsilon^2}\right)$ *columns from* **A** *by leverage score gives an* ϵ *factor subspace embedding with high probability.*



Theorem (Subspace Embedding via Sampling)

Sampling $O\left(\frac{n \log n}{\epsilon^2}\right)$ *columns from* **A** *by leverage score gives an* ϵ *factor subspace embedding with high probability.*

$$\frac{n \log n}{\epsilon^2} = \sum_i \tau(\mathbf{a}_i) \frac{\log n}{\epsilon^2}$$

Naively, computing leverage scores requires computing $(AA^T)^{-1}$, which would be difficult for a large A.

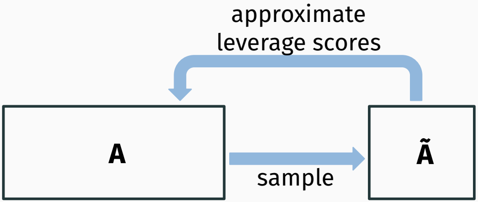
IMPORTANT NOTE

Naively, computing leverage scores requires computing $(\mathbf{A}\mathbf{A}^T)^{-1}$, which would be difficult for a large $\mathbf{A}$.

Fortunately, leverage scores are very robust – they can be estimated using very weak approximations to $\mathbf{A}$.

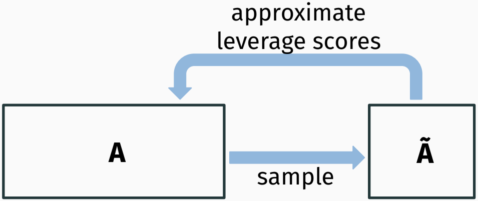
Naively, computing leverage scores requires computing $(\mathbf{A}\mathbf{A}^T)^{-1}$, which would be difficult for a large $\mathbf{A}$.

Fortunately, leverage scores are very robust – they can be estimated using very weak approximations to $\mathbf{A}$.



Naively, computing leverage scores requires computing $(\mathbf{A}\mathbf{A}^T)^{-1}$, which would be difficult for a large $\mathbf{A}$.

Fortunately, leverage scores are very robust – they can be estimated using very weak approximations to $\mathbf{A}$.



Can even be computed in a single pass over $\mathbf{A}$'s columns!

Leverage scores have been very influential, even beyond direct application to subspace embeddings.

Leverage scores have been very influential, even beyond direct application to subspace embeddings.

linear system solving, low-rank approximation, k-means clustering, convex optimization, linear programming, matrix completion, multi-label classification, spectral graph problems

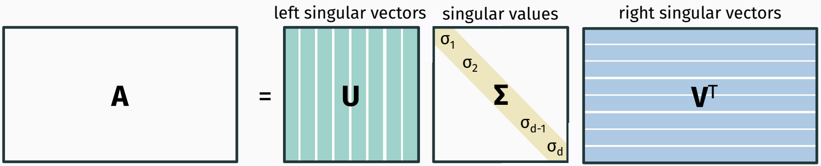
Leverage scores have been very influential, even beyond direct application to subspace embeddings.

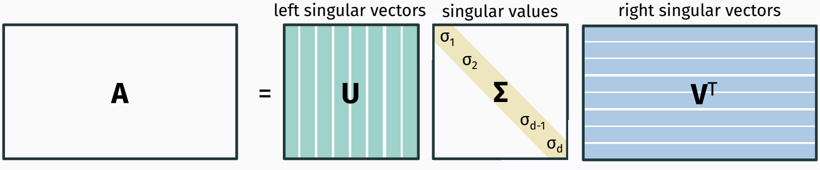
linear system solving, low-rank approximation, k-means clustering, convex optimization, linear programming, matrix completion, multi-label classification, spectral graph problems

There are many generalizations and modifications of leverage scores.

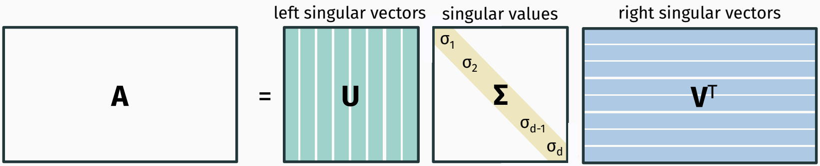
EXTENDING LEVERAGE SCORES

Extensions to low-rank problems have been especially popular.


left singular vectors
singular values
right singular vectors
A
=
U
σ1
σ2
Σ
σd-1
σd
V^T

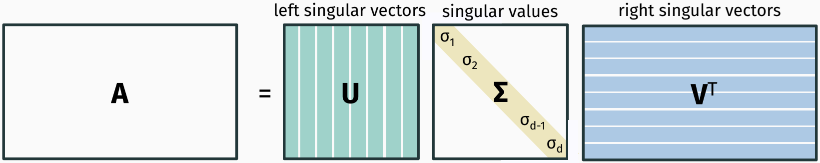


For subspace embeddings we approximate $\mathbf{A}\mathbf{A}^T = \mathbf{U}\boldsymbol{\Sigma}^2\mathbf{U}^T$.



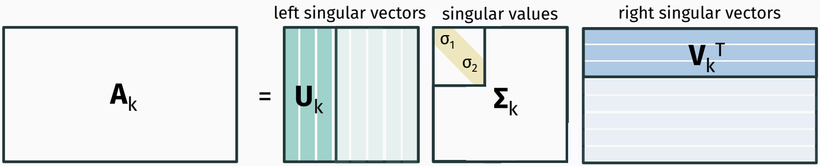
For subspace embeddings we approximate $\mathbf{A}\mathbf{A}^T = \mathbf{U}\boldsymbol{\Sigma}^2\mathbf{U}^T$.

For $\mathbf{x}^T\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\mathbf{x} \approx \mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x}$ for all $\mathbf{x}$ we need to preserve information about every singular direction/value.

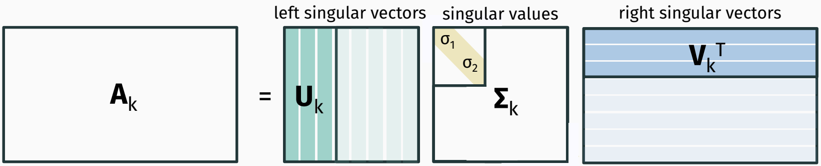


For subspace embeddings we approximate $\mathbf{A}\mathbf{A}^T = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^T$.

For $\mathbf{x}^T\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T\mathbf{x} \approx \mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x}$ for all $\mathbf{x}$ we need to preserve information about every singular direction/value. Specifically, it can be shown that $\sigma_i(\tilde{\mathbf{A}}) = (1 \pm \epsilon)\sigma_i(\mathbf{A})$



For many sketching applications, we only need $\tilde{\mathbf{A}}$ to capture information about $\mathbf{A}$'s top singular directions/values.

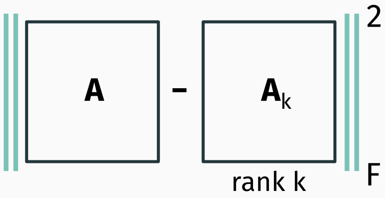


For many sketching applications, we only need $\tilde{\mathbf{A}}$ to capture information about $\mathbf{A}$'s top singular directions/values.

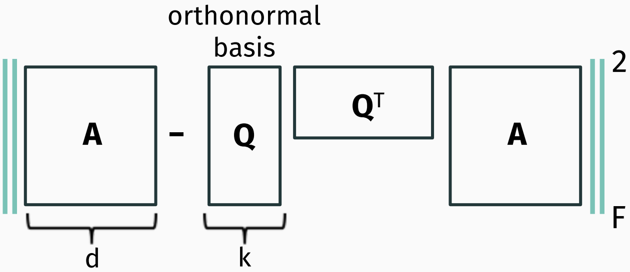
In these cases, we should be able to obtain smaller sketches – i.e. $O(k)$ instead of $O(n)$.

Find low-rank matrix close to **A**.

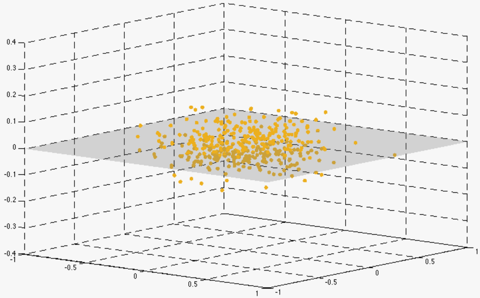
Find low-rank matrix close in Frobenius norm to **A**.



Find low-rank matrix close in Frobenius norm to **A**.



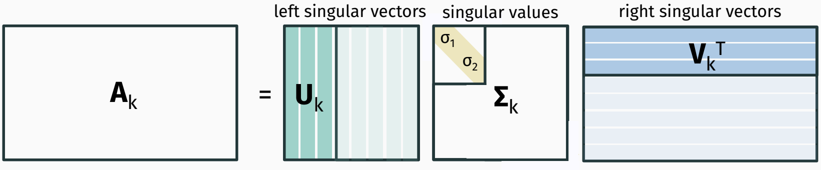
Find low-rank matrix close in Frobenius norm to $\mathbf{A}$.



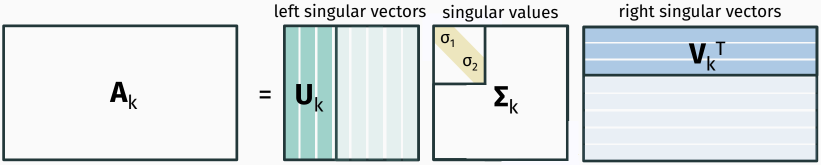
$$\|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2 = \text{sum of squared distances to hyperplane spanned by } \mathbf{Q}.$$

Without any constraints, finding the optimal rank k $\mathbf{Q}$ is equivalent to singular value decomposition:

Without any constraints, finding the optimal rank k **Q** is equivalent to singular value decomposition:



Without any constraints, finding the optimal rank k $\mathbf{Q}$ is equivalent to singular value decomposition:



$$\|\mathbf{A} - \mathbf{A}_k\|_F^2 = \|\mathbf{A} - \mathbf{U}_k\mathbf{U}_k^T\mathbf{A}\|_F^2 = \min \|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2.$$

Set $\mathbf{Q} = \mathbf{U}_k$, i.e. to the top k singular vectors of $\mathbf{A}$.

With constraints, Frobenius norm low-rank approximation captures a variety of additional interesting problems:

With constraints, Frobenius norm low-rank approximation captures a variety of additional interesting problems:

$$\min_{rank(\mathbf{Q})=k, \mathbf{Q}\in\mathcal{S}} \|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2$$

With constraints, Frobenius norm low-rank approximation captures a variety of additional interesting problems:

$$\min_{rank(\mathbf{Q})=k, \mathbf{Q}\in\mathcal{S}} \|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2$$

$\mathcal{S}$ is an arbitrary set of rank k orthonormal matrices.

With constraints, Frobenius norm low-rank approximation captures a variety of additional interesting problems:

$$\min_{rank(\mathbf{Q})=k, \mathbf{Q}\in\mathcal{S}} \|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2$$

$\mathcal{S}$ is an arbitrary set of rank k orthonormal matrices.

- nonnegative PCA

With constraints, Frobenius norm low-rank approximation captures a variety of additional interesting problems:

$$\min_{rank(\mathbf{Q})=k, \mathbf{Q}\in\mathcal{S}} \|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2$$

$\mathcal{S}$ is an arbitrary set of rank k orthonormal matrices.

- nonnegative PCA
- sparse PCA

With constraints, Frobenius norm low-rank approximation captures a variety of additional interesting problems:

$$\min_{rank(\mathbf{Q})=k, \mathbf{Q} \in \mathcal{S}} \|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2$$

$\mathcal{S}$ is an arbitrary set of rank k orthonormal matrices.

- nonnegative PCA
- sparse PCA
- k-means clustering (see slides on my website)

In either case, we need to capture information about $\mathbf{A}$'s top singular vectors only.

Two well studied guarantees for low-rank sketching.

Two well studied guarantees for low-rank sketching.

Column Subset Selection:

Find an $\tilde{\mathsf{A}}$ such that $\|\mathsf{A} - proj_{\tilde{\mathsf{A}}}(\mathsf{A})\|_F^2 \leq (1+\epsilon)\|\mathsf{A} - \mathsf{A}_k\|_F^2$.

Two well studied guarantees for low-rank sketching.

Column Subset Selection:

Find an $\tilde{\mathbf{A}}$ such that $\|\mathbf{A} - proj_{\tilde{\mathbf{A}}}(\mathbf{A})\|_F^2 \leq (1+\epsilon)\|\mathbf{A} - \mathbf{A}_k\|_F^2$.

Projection Cost Preserving Sample:

Find an $\tilde{\mathbf{A}}$ such that $\|\tilde{\mathbf{A}} - \mathbf{Q}\mathbf{Q}^T\tilde{\mathbf{A}}\|_F^2 = (1 \pm \epsilon)\|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2$ for all rank k orthonormal matrices $\mathbf{Q}$.

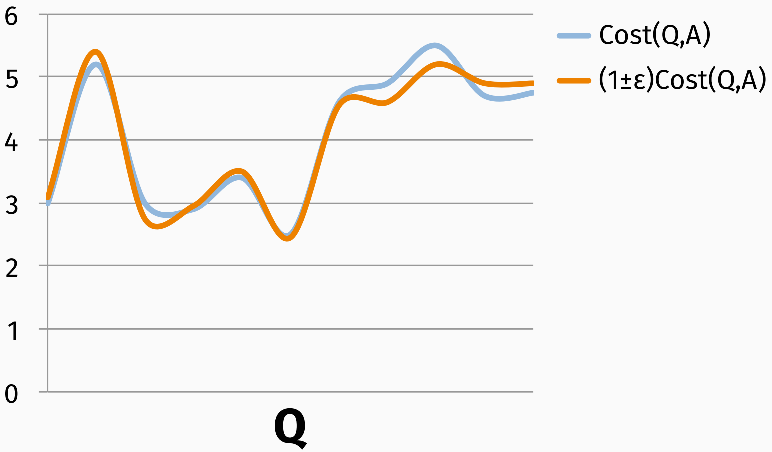
Two well studied guarantees for low-rank sketching.

Column Subset Selection:

Find an $\tilde{\mathbf{A}}$ such that $\|\mathbf{A} - proj_{\tilde{\mathbf{A}}}(\mathbf{A})\|_F^2 \leq (1+\epsilon)\|\mathbf{A} - \mathbf{A}_k\|_F^2$.

Projection Cost Preserving Sample:

Find an $\tilde{\mathbf{A}}$ such that $\|\tilde{\mathbf{A}} - \mathbf{Q}\mathbf{Q}^T\tilde{\mathbf{A}}\|_F^2 = (1 \pm \epsilon)\|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2$ for all rank k orthonormal matrices $\mathbf{Q}$.



$$\|\tilde{A} - QQ^\top \tilde{A}\|_F^2 = (1 \pm \epsilon)\|A - QQ^\top A\|_F^2$$

Subspace Embedding implies Column Subset Selection and Projection Cost Preservation.

Subspace Embedding implies Column Subset Selection and Projection Cost Preservation.

But we would get a sketch with too many samples:
$\tilde{O}(n)$ columns vs. ideally $\tilde{O}(k)$ columns.

"Low-rank leverage scores" for **column-subset selection**:

"Low-rank leverage scores" for **column-subset selection**:

$$\tau(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$$

"Low-rank leverage scores" for **column-subset selection**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\mathbf{a}_i$$

"Low-rank leverage scores" for **column-subset selection**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\mathbf{a}_i$$

- Equivalent to leverage score sampling from $\mathbf{A}_k$, but we keep the columns in $\mathbf{A}$.

"Low-rank leverage scores" for **column-subset selection**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\mathbf{a}_i$$

- Equivalent to leverage score sampling from $\mathbf{A}_k$, but we keep the columns in $\mathbf{A}$.
- Gives an approximation to $\mathbf{A}_k\mathbf{A}_k^T$, but with additional error depending on the matrix tail $\|\mathbf{A} - \mathbf{A}_k\|_F^2$.

"Low-rank leverage scores" for **column-subset selection**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\mathbf{a}_i$$

- Equivalent to leverage score sampling from $\mathbf{A}_k$, but we keep the columns in $\mathbf{A}$.
- Gives an approximation to $\mathbf{A}_k\mathbf{A}_k^T$, but with additional error depending on the matrix tail $\|\mathbf{A} - \mathbf{A}_k\|_F^2$.
- $\sum_i \tilde{\tau}(\mathbf{a}_i) = \text{rank}(\mathbf{A}_k) = k$

"Low-rank leverage scores" for **column-subset selection**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\mathbf{a}_i$$

- Equivalent to leverage score sampling from $\mathbf{A}_k$, but we keep the columns in $\mathbf{A}$.
- Gives an approximation to $\mathbf{A}_k\mathbf{A}_k^T$, but with additional error depending on the matrix tail $\|\mathbf{A} - \mathbf{A}_k\|_F^2$.
- $\sum_i \tilde{\tau}(\mathbf{a}_i) = \text{rank}(\mathbf{A}_k) = k$

[Drineas, Mahoney, Muthukrishnan '08, and Sarlós '06]

“Low-rank leverage scores” for **projection cost preservation**:

"Low-rank leverage scores" for **projection cost preservation**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\mathbf{a}_i$$

"Low-rank leverage scores" for **projection cost preservation**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T \left((\mathbf{A}_{2k}^T \mathbf{A}_k)^{-1} + \frac{k}{\|\mathbf{A} - \mathbf{A}_{2k}\|_F^2} (\mathbf{I} - \mathbf{U}_{2k}\mathbf{U}_{2k}^T) \right) \mathbf{a}_i$$

"Low-rank leverage scores" for **projection cost preservation**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T \left((\mathbf{A}_{2k}^T \mathbf{A}_k)^{-1} + \frac{k}{\|\mathbf{A} - \mathbf{A}_{2k}\|_F^2} (\mathbf{I} - \mathbf{U}_{2k}\mathbf{U}_{2k}^T) \right) \mathbf{a}_i$$

- Similar intuition, but with an extra term to capture some information about $\mathbf{A}$'s tail singular values.

"Low-rank leverage scores" for **projection cost preservation**:

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T \left((\mathbf{A}_{2k}^T \mathbf{A}_k)^{-1} + \frac{k}{\|\mathbf{A} - \mathbf{A}_{2k}\|_F^2} (\mathbf{I} - \mathbf{U}_{2k}\mathbf{U}_{2k}^T) \right) \mathbf{a}_i$$

- Similar intuition, but with an extra term to capture some information about $\mathbf{A}$'s tail singular values.

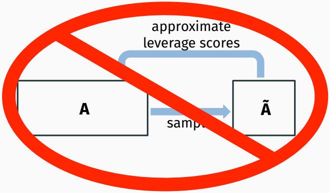
[Cohen, Elder, Musco, Musco, Persu '15]

Great, we can solve both low-rank sampling problems.

Great, we can solve both low-rank sampling problems. But...

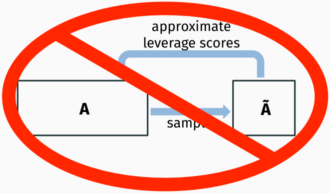
Great, we can solve both low-rank sampling problems. But...

1. The only efficient algorithms for computing low-rank leverage scores rely on other sketching techniques, often defeating the purpose of sampling to begin with.



Great, we can solve both low-rank sampling problems. But...

1. The only efficient algorithms for computing low-rank leverage scores rely on other sketching techniques, often defeating the purpose of sampling to begin with.



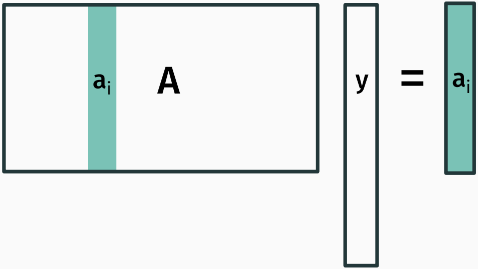
2. The scores cannot be computed in a data stream.

Single Underlying Issue:

Existing low-rank scores are not monotonic.

Single Underlying Issue:

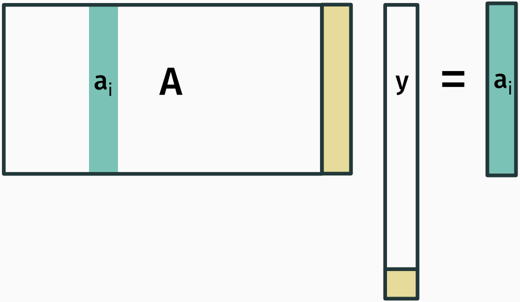
Existing low-rank scores are not monotonic.



$$\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2 \text{ such that } \mathbf{a}_i = \mathbf{A}\mathbf{y}$$

Single Underlying Issue:

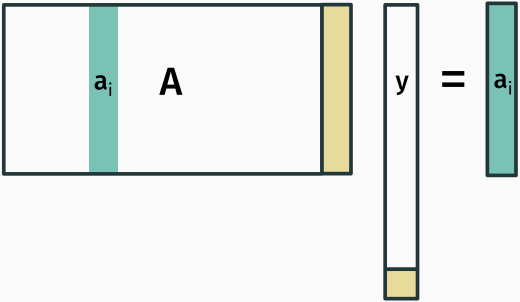
Existing low-rank scores are not monotonic.



$$\tau(\mathbf{a}_i) = \min \|\mathbf{y}\|_2^2 \text{ such that } \mathbf{a}_i = \mathbf{A}\mathbf{y}$$

Single Underlying Issue:

Existing low-rank scores are not monotonic.



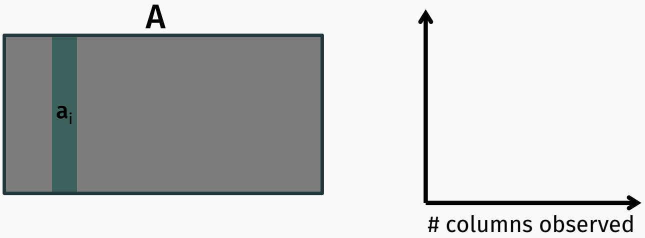
For standard leverage scores, adding a column to $\mathbf{A}$ can only decrease the importance of existing columns.

Streaming setup:

Receive columns of $\mathbf{A}$ one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].

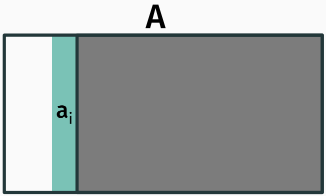
Streaming setup:

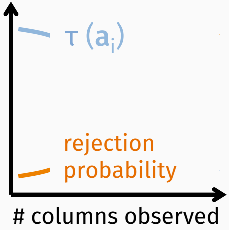
Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].



Streaming setup:

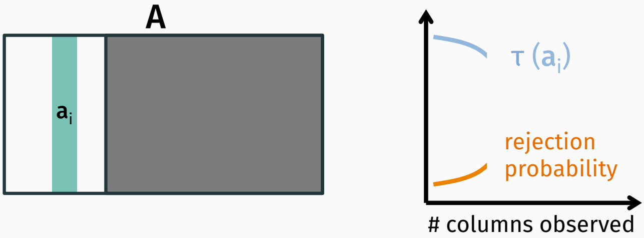
Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].





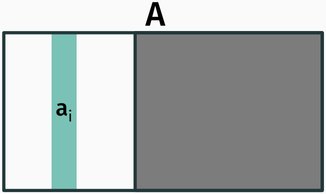
Streaming setup:

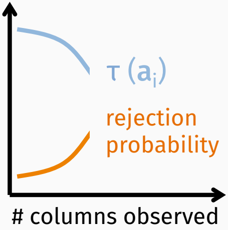
Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].



Streaming setup:

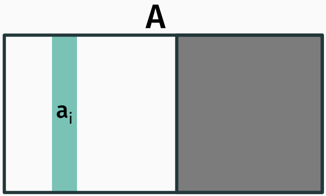
Receive columns of $\mathbf{A}$ one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].

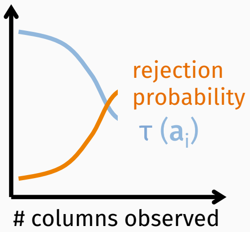




Streaming setup:

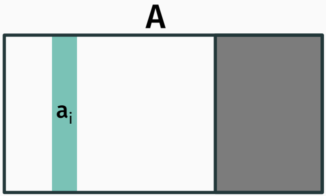
Receive columns of $\mathbf{A}$ one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].

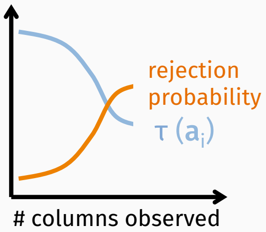




Streaming setup:

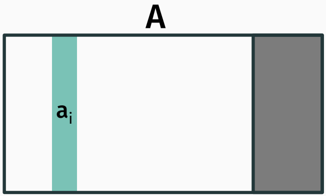
Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].

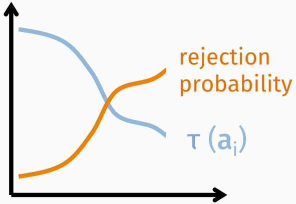




Streaming setup:

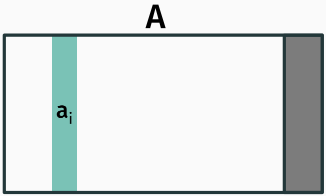
Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].

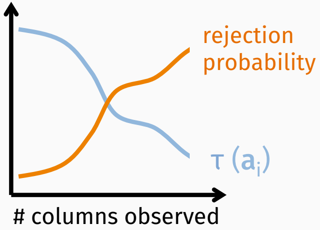




Streaming setup:

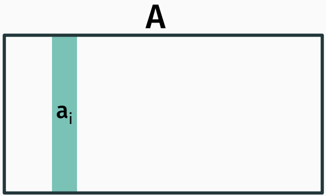
Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].

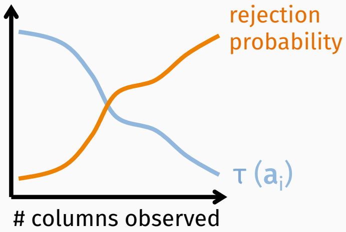




Streaming setup:

Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].

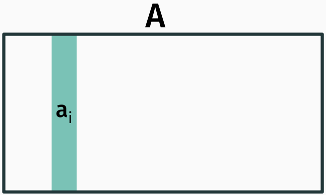


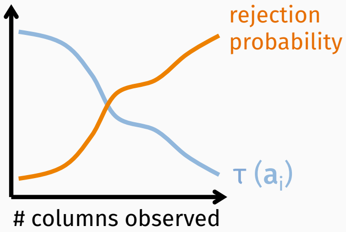


IMPORTANCE OF MONOTONICITY

Streaming setup:

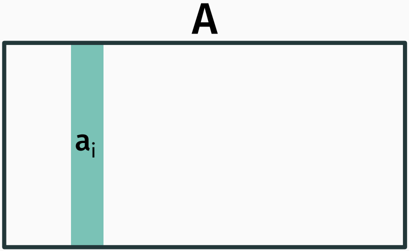
Receive columns of **A** one-by-one. Reject each with probability depending on it's (low-rank) leverage score with respect to the columns seen so far [Kelner, Levin '11].



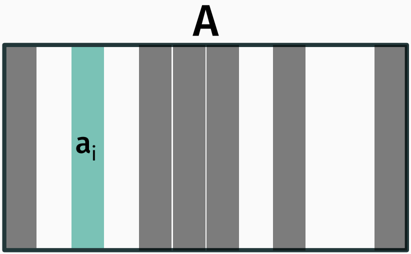


Rejection probability only decreases, so we never delete a column with too high of probability.

Iterative Leverage Score Sampling: Monotonicity is essential because it ensures that a uniform subsample of columns can at least be used to find upper bounds for leverage scores. [Cohen, Lee, Musco, Musco, Peng, Sidford '15]



Iterative Leverage Score Sampling: Monotonicity is essential because it ensures that a uniform subsample of columns can at least be used to find upper bounds for leverage scores. [Cohen, Lee, Musco, Musco, Peng, Sidford '15]



Why aren’t prior low-rank leverage scores monotonic?

.

Why aren't prior low-rank leverage scores monotonic?

They depend on $(\mathbf{A}_k \mathbf{A}_k)^{-1}$, which is inherently unstable.

SCORE INSTABILITY

Why aren't prior low-rank leverage scores monotonic?

They depend on $(\mathbf{A}_k\mathbf{A}_k)^{-1}$, which is inherently unstable.

A

1	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0
0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

$\mathbf{u}_1$

1
0
0
0
0
0

Why aren't prior low-rank leverage scores monotonic?

They depend on $(A_k A_k)^{-1}$, which is inherently unstable.

$\mathbf{A}$

1	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0
0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**2**
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

$\mathbf{u}_1$

0
1
0
0
0
0

Why aren't prior low-rank leverage scores monotonic?

They depend on $(\mathbf{A}_k\mathbf{A}_k)^{-1}$, which is inherently unstable.

$$\mathbf{A} = \begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \mathbf{u}_1 = \begin{bmatrix} 0 \\ \mathbf{1} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Adding a column could cause $\mathbf{a}_i^T(\mathbf{A}_k\mathbf{A}_k)^{-1}\mathbf{a}_i$ to drop significantly.

Here $\mathbf{a}_1^T(\mathbf{A}_1\mathbf{A}_1)^{-1}\mathbf{a}_1 \implies 0$.

How to avoid instability?

How to avoid instability?

"Soften" the existing definition of rank k leverage scores.

How to avoid instability?

"Soften" the existing definition of rank k leverage scores.

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\mathbf{a}_i$$

How to avoid instability?

"Soften" the existing definition of rank k leverage scores.

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{a}_i$$

How to avoid instability?

"Soften" the existing definition of rank k leverage scores.

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{a}_i$$

The λ-Ridge Leverage Scores of [Alaoui, Mahoney '15].

How to avoid instability?

"Soften" the existing definition of rank k leverage scores.

$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{a}_i$$

The λ-Ridge Leverage Scores of [Alaoui, Mahoney '15].

$$\sigma_i\left(\mathbf{A}^T\mathbf{A}(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\right) = \begin{cases} 1 & \text{for } i \geq k, \\ 0 & \text{for } i < k. \end{cases}$$

How to avoid instability?

"Soften" the existing definition of rank k leverage scores.

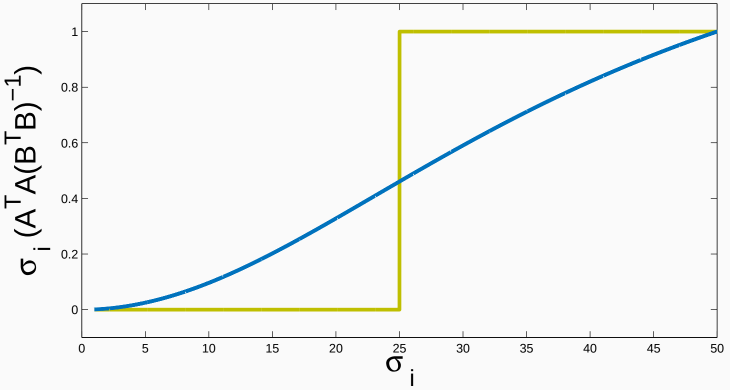
$$\tilde{\tau}(\mathbf{a}_i) = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{a}_i$$

The λ-Ridge Leverage Scores of [Alaoui, Mahoney '15].

$$\sigma_i\left(\mathbf{A}^T\mathbf{A}(\mathbf{A}_k^T\mathbf{A}_k)^{-1}\right) = \begin{cases} 1 & \text{for } i \geq k, \\ 0 & \text{for } i < k. \end{cases}$$

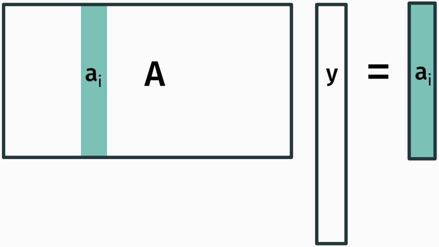
$$\sigma_i\left(\mathbf{A}^T\mathbf{A}(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\right) = \frac{\sigma_i^2}{\sigma_i^2 + \lambda}$$

Relatively “gentle” soft step:

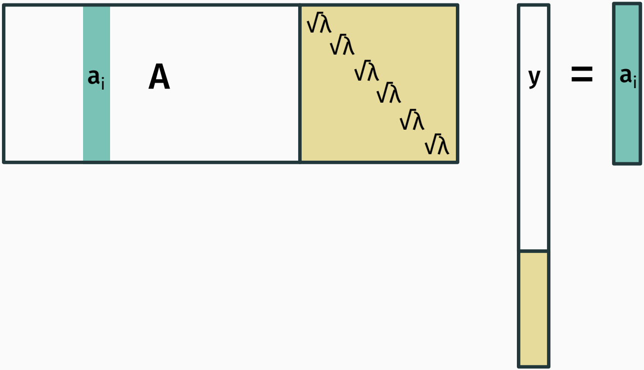


We can "wash out" the importance of columns by computing leverage scores over $\mathbf{A}$ with an identity appended:

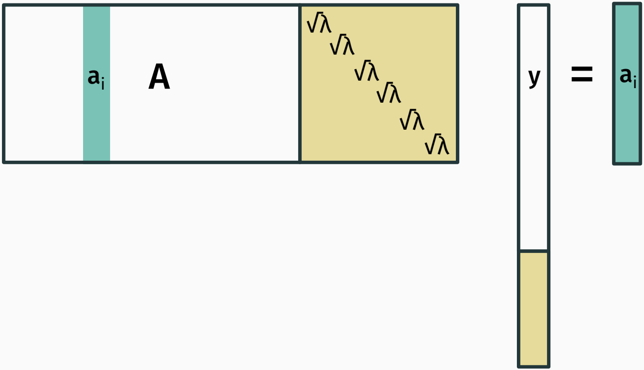
We can "wash out" the importance of columns by computing leverage scores over **A** with an identity appended:



We can "wash out" the importance of columns by computing leverage scores over **A** with an identity appended:



We can “wash out” the importance of columns by computing leverage scores over **A** with an identity appended:



Effect is weaker when $\mathbf{a}_i$ aligns with large singular vectors of **A**.

Theorem (Ridge Leverage Score Sampling)

With λ *set to* $\|\mathbf{A} - \mathbf{A}_k\|_F^2/k$, *sampling* $O(k \log k/\epsilon^2)$ *columns by ridge leverage score produces an* ϵ *error projection cost preserving sketch with high probability.*

Theorem (Ridge Leverage Score Sampling)

With λ *set to* $\|A - A_k\|_F^2/k$*, sampling* $O(k \log k/\epsilon^2)$ *columns by ridge leverage score produces an* ϵ *error projection cost preserving sketch with high probability. Sampling* $O(k \log k/\epsilon)$ *columns produces an* ϵ *error column subset.*

Theorem (Ridge Leverage Score Sampling)

With λ set to $\|\mathbf{A} - \mathbf{A}_k\|_F^2/k$, sampling $O(k \log k/\epsilon^2)$ columns by ridge leverage score produces an ϵ error projection cost preserving sketch with high probability. Sampling $O(k \log k/\epsilon)$ columns produces an ϵ error column subset.

Furthermore, $(\|\mathbf{A} - \mathbf{A}_k\|_F^2/k)$-ridge leverage scores are monotonic with respect to column additions.

Since $\lambda = \|\mathbf{A} - \mathbf{A}_k\|_F^2$ can only increase as columns are added to $\mathbf{A}$, this perspective immediately implies that ridge leverage score are monotonic.

With λ set to $\|\mathbf{A} - \mathbf{A}_k\|_F^2/k$, sampling by ridge leverage score produces a sketch $\tilde{\mathbf{A}}$ such that:

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\frac{\|\mathbf{A} - \mathbf{A}_k\|_F^2}{k}\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\frac{\|\mathbf{A} - \mathbf{A}_k\|_F^2}{k}\mathbf{I}$$

INTERMEDIATE RESULT

With λ set to $\|\mathbf{A} - \mathbf{A}_k\|_F^2/k$, sampling by ridge leverage score produces a sketch $\tilde{\mathbf{A}}$ such that:

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\frac{\|\mathbf{A} - \mathbf{A}_k\|_F^2}{k}\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\frac{\|\mathbf{A} - \mathbf{A}_k\|_F^2}{k}\mathbf{I}$$

Multiplicative error of a **subspace embedding**.

With λ set to $\|\mathbf{A}-\mathbf{A}_k\|_F^2/k$, sampling by ridge leverage score produces a sketch $\tilde{\mathbf{A}}$ such that:

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}$$

Multiplicative error of a **subspace embedding**.

Additive error of a **Frequent Directions sketch** [Ghashami, Liberty, Phillips, Woodruff].

With λ set to $\|\mathbf{A} - \mathbf{A}_k\|_F^2/k$, sampling by ridge leverage score produces a sketch $\tilde{\mathbf{A}}$ such that:

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\frac{\|\mathbf{A} - \mathbf{A}_k\|_F^2}{k}\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\frac{\|\mathbf{A} - \mathbf{A}_k\|_F^2}{k}\mathbf{I}$$

Multiplicative error of a **subspace embedding**.

Additive error of a **Frequent Directions sketch** [Ghashami, Liberty, Phillips, Woodruff].

Both are known to give projection cost preserving sketches. Handling both errors simultaneously is tedious, but not hard.

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}$$

MIXED ERROR BOUND $\rightarrow$ PROJECTION COST PRESERVING SKETCH

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A - A_k\|_F^2}{k}I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A - A_k\|_F^2}{k}I$$

MIXED ERROR BOUND → PROJECTION COST PRESERVING SKETCH

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A-A_k\|_F^2}{k}I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A-A_k\|_F^2}{k}I$$

$$\|\tilde{A} - QQ^T\tilde{A}\|_F^2 = (1 \pm \epsilon)\|A - QQ^TA\|_F^2$$

MIXED ERROR BOUND → PROJECTION COST PRESERVING SKETCH

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A - A_k\|_F^2}{k}I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A - A_k\|_F^2}{k}I$$

$$\|(I - QQ^T)\tilde{A}\|_F^2 = (1 \pm \epsilon)\|(I - QQ^T)A\|_F^2$$

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A-A_k\|_F^2}{k}I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A-A_k\|_F^2}{k}I$$

$$\|(I-QQ^T)\tilde{A}\|_F^2 = (1\pm\epsilon)\|(I-QQ^T)A\|_F^2$$

Sum of vector products with $\tilde{A}$. Each preserved to within a $(1\pm\epsilon)$ factor, so the entire sum is as well.

MIXED ERROR BOUND $\rightarrow$ PROJECTION COST PRESERVING SKETCH

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A - A_k\|_F^2}{k} I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A - A_k\|_F^2}{k} I$$

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A - A_k\|_F^2}{k} I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A - A_k\|_F^2}{k} I$$

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A - A_k\|_F^2}{k}I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A - A_k\|_F^2}{k}I$$

Intuition:

Dealing with rank k operators (Q is rank k), so we only pay the additive error k times.

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A - A_k\|_F^2}{k} I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A - A_k\|_F^2}{k} I$$

Intuition:

Dealing with rank k operators (Q is rank k), so we only pay the additive error k times.

$$\text{total additive error} = k \cdot \epsilon\frac{\|A - A_k\|_F^2}{k}$$

$$(1-\epsilon)\tilde{A}\tilde{A}^T - \epsilon\frac{\|A - A_k\|_F^2}{k}I \preceq AA^T \preceq (1+\epsilon)\tilde{A}\tilde{A}^T + \epsilon\frac{\|A - A_k\|_F^2}{k}I$$

Intuition:

Dealing with rank k operators (Q is rank k), so we only pay the additive error k times.

$$\text{total additive error} = k \cdot \epsilon\frac{\|A - A_k\|_F^2}{k}$$
$$= \epsilon\|A - A_k\|_F^2$$

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}$$

Intuition:

Dealing with rank k operators ($\mathbf{Q}$ is rank k), so we only pay the additive error k times.

$$\begin{aligned}\text{total additive error} &= k \cdot \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k} \\ &= \epsilon\|\mathbf{A}-\mathbf{A}_k\|_F^2 \\ &\leq \epsilon\|\mathbf{A}-\mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F^2\end{aligned}$$

Since $\mathbf{A}_k$ is a better low-rank approximation than any $\mathbf{Q}\mathbf{Q}^T\mathbf{A}$.

$$(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}$$

Proof follows directly from our “appending an identity” view!

Proof:

Proof:

1. Leverage score sampling clearly works if we set $p_i > \frac{\log n}{\epsilon}\tau_i$.

Proof:

1. Leverage score sampling clearly works if we set $p_i > \frac{\log n}{\epsilon}\tau_i$.
2. Take identity columns with probability one, everything else with leverage score probabilities.

Proof:

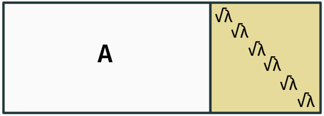
1. Leverage score sampling clearly works if we set $p_i > \frac{\log n}{\epsilon}\tau_i$.
2. Take identity columns with probability one, everything else with leverage score probabilities.
3. Obtain a sketch $\mathbf{B} = [\tilde{\mathbf{A}}, \sqrt{\lambda}\mathbf{I}]$ satisfying:
   $(1-\epsilon)\mathbf{B}\mathbf{B}^T \preceq \mathbf{A}\mathbf{A}^T + \lambda\mathbf{I} \preceq (1+\epsilon)\mathbf{B}\mathbf{B}^T$

Proof:

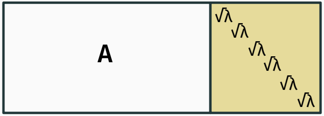
1. Leverage score sampling clearly works if we set $p_i > \frac{\log n}{\epsilon}\tau_i$.
2. Take identity columns with probability one, everything else with leverage score probabilities.
3. Obtain a sketch $\mathbf{B} = [\tilde{\mathbf{A}}, \sqrt{\lambda}\mathbf{I}]$ satisfying: $(1-\epsilon)\mathbf{B}\mathbf{B}^T \preceq \mathbf{A}\mathbf{A}^T + \lambda\mathbf{I} \preceq (1+\epsilon)\mathbf{B}\mathbf{B}^T$
4. $(1-\epsilon)(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \lambda\mathbf{I}) \preceq \mathbf{A}\mathbf{A}^T + \lambda\mathbf{I} \preceq (1+\epsilon)(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \lambda\mathbf{I})$

Proof:

1. Leverage score sampling clearly works if we set $p_i > \frac{\log n}{\epsilon}\tau_i$.
2. Take identity columns with probability one, everything else with leverage score probabilities.
3. Obtain a sketch $\mathbf{B} = [\tilde{\mathbf{A}}, \sqrt{\lambda}\mathbf{I}]$ satisfying: $(1-\epsilon)\mathbf{B}\mathbf{B}^T \preceq \mathbf{A}\mathbf{A}^T + \lambda\mathbf{I} \preceq (1+\epsilon)\mathbf{B}\mathbf{B}^T$
4. $(1-\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T - \epsilon\lambda\mathbf{I} \preceq \mathbf{A}\mathbf{A}^T \preceq (1+\epsilon)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^T + \epsilon\lambda\mathbf{I}$



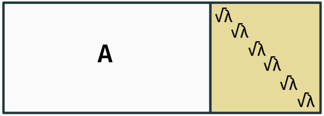
Number of columns sampled to form $\tilde{\mathbf{A}}$ depends on sum of leverage scores, outside of the identity columns.



Number of columns sampled to form $\tilde{\mathbf{A}}$ depends on sum of leverage scores, outside of the identity columns.

$$\sum_{i=1}^{d} \tilde{\tau}(\mathbf{a}_i) = \operatorname{tr}\left(\mathbf{A}^T \left(\mathbf{A}^T\mathbf{A} + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}\right)^{-1} \mathbf{A}\right)$$

MIXED ERROR PROOF

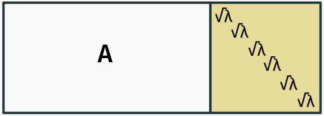


Number of columns sampled to form $\tilde{\mathbf{A}}$ depends on sum of leverage scores, outside of the identity columns.

$$\sum_{i=1}^{d} \tilde{\tau}(\mathbf{a}_i) = \operatorname{tr}\left(\mathbf{A}^T \left(\mathbf{A}^T\mathbf{A} + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}\right)^{-1} \mathbf{A}\right)$$

$$= \sum_{i=1}^{d} \sigma_i \left(\mathbf{A}^T \left(\mathbf{A}^T\mathbf{A} + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}\right)^{-1} \mathbf{A}\right)$$

MIXED ERROR PROOF

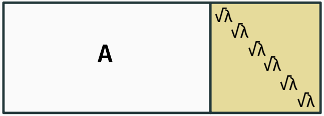


Number of columns sampled to form $\tilde{A}$ depends on sum of leverage scores, outside of the identity columns.

$$\sum_{i=1}^{d} \tilde{\tau}(a_i) = \operatorname{tr}\left(A^T\left(A^TA + \frac{\|A - A_k\|_F^2}{k} I\right)^{-1} A\right)$$

$$= \sum_{i=1}^{d} \frac{\sigma_i(A)}{\sigma_i(A) + \frac{\|A - A_k\|_F^2}{k}}$$

MIXED ERROR PROOF



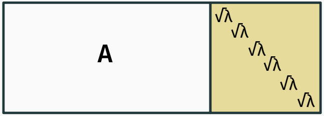
Number of columns sampled to form $\tilde{\mathbf{A}}$ depends on sum of leverage scores, outside of the identity columns.

$$\sum_{i=1}^{d} \tilde{\tau}(\mathbf{a}_i) = \operatorname{tr}\left(\mathbf{A}^T\left(\mathbf{A}^T\mathbf{A} + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}\right)^{-1}\mathbf{A}\right)$$

$$= \sum_{i=1}^{d} \frac{\sigma_i(\mathbf{A})}{\sigma_i(\mathbf{A}) + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

$$\leq k + \sum_{i=k+1}^{d} \frac{\sigma_i(\mathbf{A})}{\sigma_i(\mathbf{A}) + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

MIXED ERROR PROOF



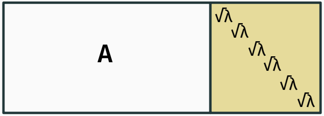
Number of columns sampled to form $\tilde{\mathbf{A}}$ depends on sum of leverage scores, outside of the identity columns.

$$\sum_{i=1}^{d} \tilde{\tau}(\mathbf{a}_i) = \operatorname{tr}\left(\mathbf{A}^T\left(\mathbf{A}^T\mathbf{A} + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}\right)^{-1}\mathbf{A}\right)$$

$$= \sum_{i=1}^{d} \frac{\sigma_i(\mathbf{A})}{\sigma_i(\mathbf{A}) + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

$$\leq k + \sum_{i=k+1}^{d} \frac{\sigma_i(\mathbf{A})}{\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

MIXED ERROR PROOF



Number of columns sampled to form $\tilde{\mathbf{A}}$ depends on sum of leverage scores, outside of the identity columns.

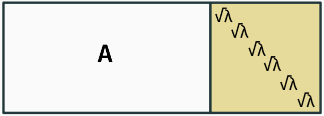
$$\sum_{i=1}^{d} \tilde{\tau}(\mathbf{a}_i) = \text{tr}\left(\mathbf{A}^T\left(\mathbf{A}^T\mathbf{A} + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}\right)^{-1}\mathbf{A}\right)$$

$$= \sum_{i=1}^{d} \frac{\sigma_i(\mathbf{A})}{\sigma_i(\mathbf{A}) + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

$$\leq k + \sum_{i=k+1}^{d} \frac{\sigma_i(\mathbf{A})}{\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

$$= k + k$$

MIXED ERROR PROOF



Number of columns sampled to form $\tilde{\mathbf{A}}$ depends on sum of leverage scores, outside of the identity columns.

$$\sum_{i=1}^{d} \tilde{\tau}(\mathbf{a}_i) = \operatorname{tr}\left(\mathbf{A}^T\left(\mathbf{A}^T\mathbf{A} + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}\mathbf{I}\right)^{-1}\mathbf{A}\right)$$

$$= \sum_{i=1}^{d} \frac{\sigma_i(\mathbf{A})}{\sigma_i(\mathbf{A}) + \frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

$$\leq k + \sum_{i=k+1}^{d} \frac{\sigma_i(\mathbf{A})}{\frac{\|\mathbf{A}-\mathbf{A}_k\|_F^2}{k}}$$

$$= k + k = O(k).$$

Proving the column subset selection result requires a bit of additional work, but otherwise the rest of our paper focus on two main applications of monotonicity:

Proving the column subset selection result requires a bit of additional work, but otherwise the rest of our paper focus on two main applications of monotonicity:

1. The first $\text{nnz}(\mathbf{A})$ time low-rank approximation algorithm based on iterative column sampling.

Proving the column subset selection result requires a bit of additional work, but otherwise the rest of our paper focus on two main applications of monotonicity:

1. The first $\text{nnz}(\mathbf{A})$ time low-rank approximation algorithm based on iterative column sampling.
2. Single pass algorithms for ridge leverage score sampling whose memory requirements do not increase with d.

ADDITIONAL RESULTS

Proving the column subset selection result requires a bit of additional work, but otherwise the rest of our paper focus on two main applications of monotonicity:

1. The first nnz($\mathbf{A}$) time low-rank approximation algorithm based on iterative column sampling.
2. Single pass algorithms for ridge leverage score sampling whose memory requirements do not increase with d.

Please checkout the arXiv preprint!