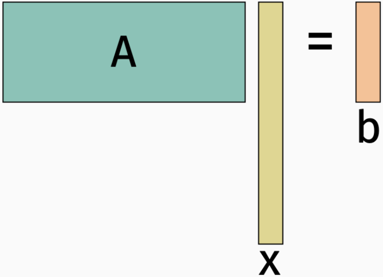
CS-GY 6763: Lecture 13
Finish Sparse Recovery and Compressed Sensing, Introduction to Spectral Sparsification

NYU Tandon School of Engineering, Prof. Christopher Musco

- Design a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $m < n$, $\mathbf{b} \in \mathbb{R}^m$.
- "Measure" $\mathbf{b} = \mathbf{A}\mathbf{x}$ for some k-sparse $\mathbf{x} \in \mathbb{R}^n$.



- Recover $\mathbf{x}$ from $\mathbf{b}$.

Sample complexity: Can achieve $m = O(k \log n)$ or similar.

- Usually corresponds to some application-dependent cost (eg. length of time to acquire MRI, space complexity for heavy hitters problem)

Computational complexity: Naive methods take $O(n^k)$ time to recover k-sparse $\mathbf{x}$ from $\mathbf{b}$.

Typically design $\mathbf{A}$ with as few rows as possible that fulfills some desired property.

- $\mathbf{A}$ has Kruskal rank r. All sets of r columns in $\mathbf{A}$ are linearly independent.
  - Recover vectors $\mathbf{x}$ with sparsity $k = r/2$.
- $\mathbf{A}$ is μ-incoherent. $|\mathbf{A}_i^T \mathbf{A}_j| \leq \mu \|\mathbf{A}_i\|_2 \|\mathbf{A}_j\|_2$ for all columns $\mathbf{A}_i, \mathbf{A}_j$, $i \neq j$.
  - Recover vectors $\mathbf{x}$ with sparsity $k = 1/\mu$.

  $\mathbf{A}$ obeys the (q, ϵ)-Restricted Isometry Property.
  - Recover vectors $\mathbf{x}$ with sparsity $k = O(q)$.

Definition ((q, ϵ)-Restricted Isometry Property)

A matrix $\mathbf{A}$ satisfies (q, ϵ)-RIP if, for all $\mathbf{x}$ with $\|\mathbf{x}\|_0 \leq q$,

$$(1-\epsilon)\|\mathbf{x}\|_2^2 \leq \|\mathbf{A}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{x}\|_2^2.$$

Argued this holds for random matrices (JL matrices) and subsampled Fourier matrices with roughly $m = O\left(\frac{k \log n}{\epsilon^2}\right)$ rows.

Theorem (ℓ_0-minimization)

Suppose we are given $\mathbf{A} \in \mathbb{R}^{m \times n}$ *and* $\mathbf{b} = \mathbf{A}\mathbf{x}$ *for an unknown* k*-sparse* $\mathbf{x} \in \mathbb{R}^n$*. If* $\mathbf{A}$ *is* $(2k, \epsilon)$*-RIP for any* $\epsilon < 1$ *then* $\mathbf{x}$ *is the* *unique minimizer of:*

$$\min \|\mathbf{z}\|_0 \qquad \textit{subject to} \qquad \mathbf{A}\mathbf{z} = \mathbf{b}.$$

- Establishes that information theoretically we can recover $\mathbf{x}$ in $O(n^k)$ time from $O(k \log n)$ measurements.

Convex relaxation of the ℓ_0 minimization problem:

Problem (Basis Pursuit, i.e. ℓ_1 minimization.)

$$\min_{\mathbf{z}} \|\mathbf{z}\|_1 \qquad \textit{subject to} \qquad \mathbf{A}\mathbf{z} = \mathbf{b}.$$

- Objective is convex.
- Optimizing over convex set.

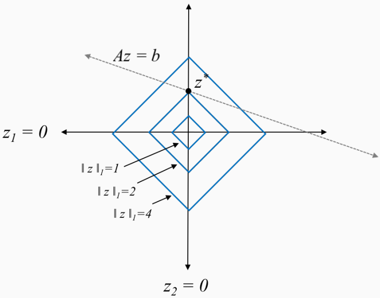
Theorem

If $\mathbf{A}$ *is* $(3k, \epsilon)$*-RIP for* $\epsilon < .17$ *and* $\|\mathbf{x}\|_0 = k$*, then* $\mathbf{x}$ *is the unique optimal solution of the Basis Pursuit LP).*

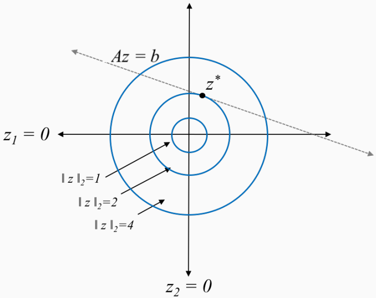
Two surprising things about this result:

- Exponentially improve computational complexity with only a constant factor overhead in measurement complexity.
- Typical "relax-and-round" algorithm, but rounding is not even necessary! Just return the solution of the relaxed problem.

Suppose **A** is 2×1, so **b** is just a scalar and **x** is a 2-dimensional vector.



Vertices of level sets of ℓ_1 norm correspond to sparse solutions.



This is not the case e.g. for the ℓ_2 norm.

Theorem

If $\mathbf{A}$ is $(3k, \epsilon)$-RIP for $\epsilon < .17$ and $\|\mathbf{x}\|_0 = k$, then $\mathbf{x}$ is the unique optimal solution of the Basis Pursuit LP).

Similar proof to ℓ_0 minimization:

- By way of contradiction, assume $\mathbf{x}$ is not the optimal solution. Then there exists some non-zero Δ such that:
  - $\|\mathbf{x} + \Delta\|_1 \leq \|\mathbf{x}\|_1$
  - $\mathbf{A}(\mathbf{x} + \Delta) = \mathbf{A}\mathbf{x}$. I.e. $\mathbf{A}\Delta = 0$.

Difference is that we can no longer assume that Δ is sparse.

We will argue that Δ is approximately sparse.

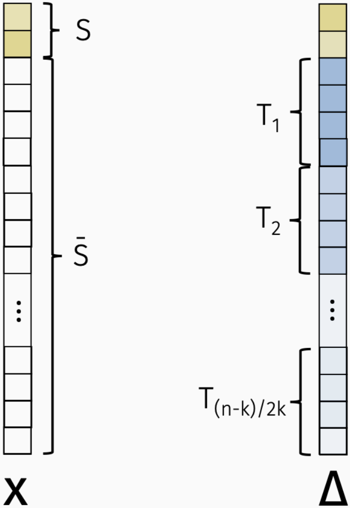
First tool:

For any q-sparse vector $\mathbf{w}$, $\|\mathbf{w}\|_2 \leq \|\mathbf{w}\|_1 \leq \sqrt{q}\|\mathbf{w}\|_2$

Second tool:

For any norm and vectors $\mathbf{a}, \mathbf{b}$, $\|\mathbf{a}+\mathbf{b}\| \geq \|\mathbf{a}\| - \|\mathbf{b}\|$

Some definitions:



Claim 1: $\|\Delta_S\|_1 \geq \|\Delta_{\bar{S}}\|_1$

Claim 2: $\|\Delta_S\|_2 \geq \sqrt{2}\sum_{j\geq 2}\|\boldsymbol{\Delta}_{T_j}\|_2$:

$$\|\boldsymbol{\Delta}_s\|_2 \geq \frac{1}{\sqrt{k}}\|\boldsymbol{\Delta}_S\|_1 \geq \frac{1}{\sqrt{k}}\|\boldsymbol{\Delta}_{\bar{S}}\|_1 = \frac{1}{\sqrt{k}}\sum_{j\geq 1}\|\boldsymbol{\Delta}_{T_j}\|_1.$$

Claim: $\|\boldsymbol{\Delta}_{T_j}\|_1 \geq \sqrt{2k}\|\boldsymbol{\Delta}_{T_{j+1}}\|_2$

Finish up proof by contradiction: Recall that $\mathbf{A}$ is assumed to have the $(3k, \epsilon)$ RIP property.

$$0 = \|\mathbf{A}\boldsymbol{\Delta}\|_2 \geq \|\mathbf{A}\boldsymbol{\Delta}_{S\cup T_1}\|_2 - \sum_{j\geq 2} \|\mathbf{A}\boldsymbol{\Delta}_{T_j}\|_2$$

A lot of interest in developing even faster algorithms that avoid using the "heavy hammer" of linear programming and run in even faster than $O(n^{3.5})$ time.

- **Iterative Hard Thresholding**: Looks a lot like projected gradient descent. Solve $\min_{\mathbf{z}} \|\mathbf{Az} - \mathbf{b}\|$ with gradient descent while continually projecting z back to the set of k-sparse vectors. Runs in time $\sim O(nk \log n)$ for Gaussian measurement matrices and $O(n \log n)$ for subsampled Fourer matrices.
- Other "first order" type methods: Orthogonal Matching Pursuit, CoSaMP, Subspace Pursuit, etc.

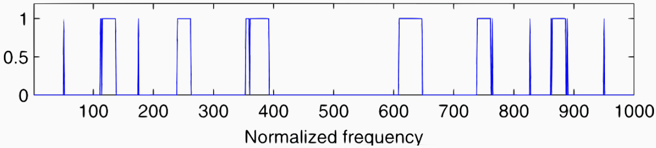
When $\mathbf{A}$ is a subsampled Fourier matrix, there are now methods that run in $O(k \log^c n)$ time [Hassanieh, Indyk, Kapralov, Katabi, Price, Shi, etc. 2012+].

Hold up...

Corollary: When $\mathbf{x}$ is k-sparse, we can compute the inverse Fourier transform $\mathbf{F}^*\mathbf{F}\mathbf{x}$ of $\mathbf{F}\mathbf{x}$ in $O(k \log^c n)$ time!

- Randomly subsample $\mathbf{F}\mathbf{x}$.
- Feed that input into our sparse recovery algorithm to extract $\mathbf{x}$.

Fourier and inverse Fourier transforms in sublinear time when the output is sparse.



Applications in: Wireless communications, GPS, protein imaging, radio astronomy, etc. etc.

A LITTLE ABOUT MY RESEARCH

Theorem (Subspace Embedding)

Let $\mathbf{A} \in \mathbb{R}^{n \times d}$ *be a matrix. If* $\mathbf{\Pi} \in \mathbb{R}^{m \times n}$ *is chosen from any distribution* $\mathcal{D}$ *satisfying the Distributional JL Lemma, then with probability* $1 - \delta$,

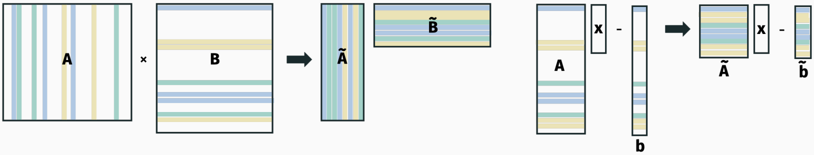
$$(1-\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2 \leq \|\mathbf{\Pi}\mathbf{A}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2$$

for <u>*all*</u> $\mathbf{x} \in \mathbb{R}^d$, *as long as* $m = O\left(\frac{d+\log(1/\delta)}{\epsilon^2}\right)$.

Implies regression result, and more.

Example: The any singular value $\tilde{\sigma}_i$ of $\mathbf{\Pi}\mathbf{A}$ is a $(1 \pm \epsilon)$ approximation to the true singular value σ_i of $\mathbf{B}$.

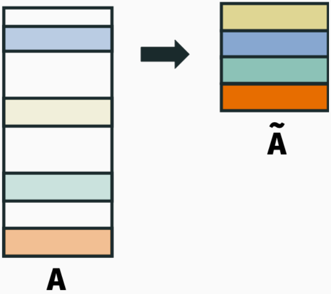
Recurring research interest: Replace random projection methods with random sampling methods. Prove that for essentially all problems of interest, can obtain same asymptotic runtimes.



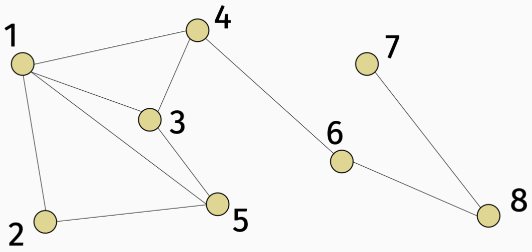
Sampling has the added benefit of preserving matrix sparsity or structure, and can be applied in a wider variety of settings where random projections are too expensive.

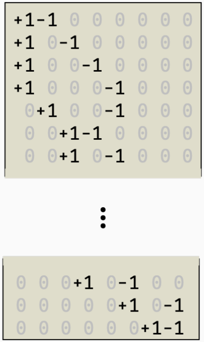
Goal: Can we use sampling to obtain subspace embeddings? I.e. for a given $\mathbf{A}$ find $\tilde{\mathbf{A}}$ whose rows are a (weighted) subset of rows in $\mathbf{A}$ and:

$$(1-\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2.$$



Let $\mathbf{B}$ be the edge-vertex incidence matrix of a graph G with vertex set V, $|V| = d$. Recall that $\mathbf{B}^T\mathbf{B} = \mathbf{L}$.

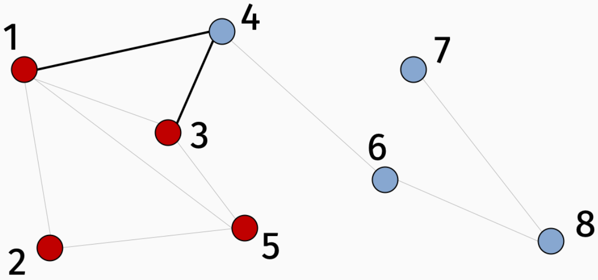




B

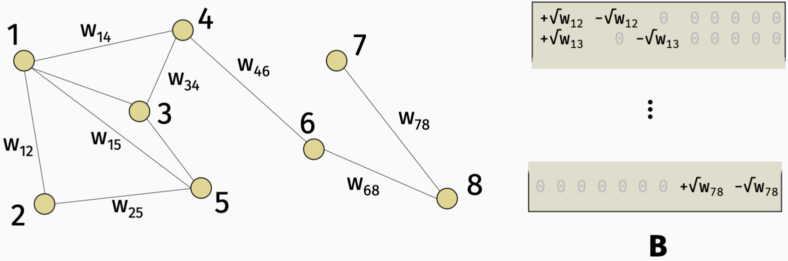
Recall that if $\mathbf{x} \in \{-1, 1\}^n$ is the cut indicator vector for a cut S in the graph, then $\frac{1}{4}\|\mathbf{B}\mathbf{x}\|_2^2 = \mathbf{cut}(S, V \setminus S)$.

$$\mathbf{x} = [1, 1, 1, -1, 1, -1, -1, -1]$$



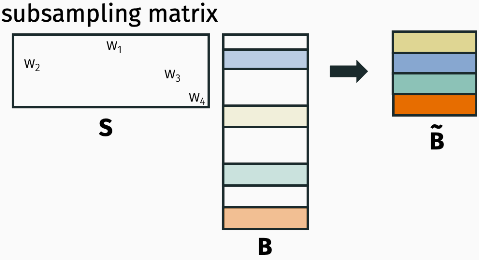
$\mathbf{x} \in \{-1, 1\}^d$ is the cut indicator vector for a cut S in the graph, then $\frac{1}{4}\|\mathbf{B}\mathbf{x}\|_2^2 = \text{cut}(S, V \setminus S)$

Extends to weighted graphs, as long as square root of weights is included in $\mathbf{B}$. Still have the $\mathbf{B}^T\mathbf{B} = \mathbf{L}$.



And still have that if $\mathbf{x} \in \{-1, 1\}^d$ is the cut indicator vector for a cut S in the graph, then $\frac{1}{4}\|\mathbf{B}\mathbf{x}\|_2^2 = \mathrm{cut}(S, V \setminus S)$.

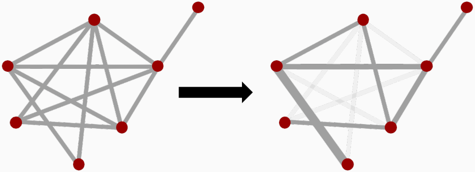
Goal: Approximate $\mathbf{B}$ by a weighted subsample. I.e. by $\tilde{\mathbf{B}}$ with $m \ll |E|$ rows, each of which is a scaled copy of a row from $\mathbf{B}$.



Natural goal: $\tilde{\mathbf{B}}$ is a subspace embedding for $\mathbf{B}$. In other words, $\tilde{\mathbf{B}}$ has $\approx O(d)$ rows and for all $\mathbf{x}$,

$$(1-\epsilon)\|\mathbf{B}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{B}}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{B}\mathbf{x}\|_2^2.$$

$\tilde{\mathbf{B}}$ is itself an edge-vertex incidence matrix for some sparser graph $\tilde{G}$, which preserves many properties about G! $\tilde{G}$ is called a spectral sparsifier for G.



For example, we have that for any set S,

$$(1-\epsilon)\,\mathbf{cut}_G(S, V\setminus S) \leq \mathbf{cut}_{\tilde{G}}(S, V\setminus S) \leq (1+\epsilon)\,\mathbf{cut}_G(S, V\setminus S).$$

So $\tilde{G}$ can be used in place of G in solving e.g. max/min cut problems, balanced cut problems, etc.

In contrast $\mathbf{\Pi B}$ would look nothing like an edge-vertex incidence matrix if $\mathbf{\Pi}$ is a JL matrix.

HISTORY OF SPECTRAL SPARSIFICATION

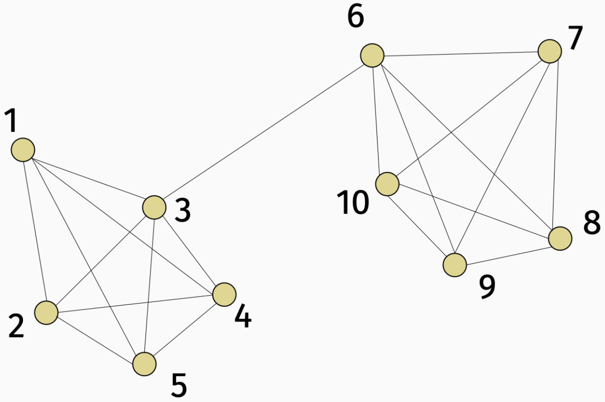
Spectral sparsifiers were introduced in 2004 by Spielman and Teng in an influential paper on faster algorithms for solving Laplacian linear systems.

- Generalize the cut sparsifiers of Benczur, Karger '96.
- Further developed in work by Spielman, Srivastava + Batson, '08.
- Have had huge influence in algorithms, and other areas of mathematics – this line of work lead to the 2013 resolution of the Kadison-Singer problem in functional analysis by Marcus, Spielman, Srivastava.

Rest of class: Learn about an important random sampling algorithm for constructing spectral sparsifiers, and subspace embeddings for matrices more generally.

Goal: Find $\tilde{\mathbf{A}}$ such that $\|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 = (1 \pm \epsilon)\|\mathbf{A}\mathbf{x}\|_2^2$ for all $\mathbf{x}$.

Possible Approach: Construct $\tilde{\mathbf{A}}$ by uniformly sampling rows from $\mathbf{A}$.



Can check that this approach fails even for the special case of a graph vertex-edge incidence matrix.

Key idea: Importance sampling. Select some rows with higher probability.

Suppose $\mathbf{A}$ has n rows $\mathbf{a}_1 \ldots, \mathbf{a}_n$. Let $p_1, \ldots, p_n \in [0, 1]$ be sampling probabilities. Construct $\tilde{\mathbf{A}}$ as follows:

- For $i = 1, \ldots, n$
  - Select $\mathbf{a}_i$ with probability p_i.
  - If $\mathbf{a}_i$ is selected, add the scaled row $\frac{1}{\sqrt{p_i}}\mathbf{a}_i$ to $\tilde{A}$.

Remember, ultimately want that $\|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 = (1 \pm \epsilon)\|\mathbf{A}\mathbf{x}\|_2^2$ for all $\mathbf{x}$.

Claim 1: $\mathbb{E}[\|\tilde{\mathbf{A}}\mathbf{x}\|_2^2] = \|\mathbf{A}\mathbf{x}\|_2^2$.

Claim 2: Expected number of rows in $\tilde{\mathbf{A}}$ is $\sum_{i=1}^{n} p_i$.

How should we choose the probabilities $p_1, \ldots, p_n$?

1. Introduce the idea of row **leverage scores**.
2. Motivate why these scores make for good sampling probabilities.
3. Prove that sampling with probabilities proportional to these scores yields a subspace embedding (or a spectral sparsifier) with a near optimal number of rows.

Let $\mathbf{a}_1, \ldots, \mathbf{a}_n$ be $\mathbf{A}$'s rows. We define the **statistical leverage score** τ_i of row $\mathbf{a}_i$ as:

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

We will show that τ_i is a natural importance measure for each row in $\mathbf{A}$.

We have that $\tau_i \in [0, 1]$ and $\sum_{i=1}^n \tau_i = d$ if $\mathbf{A}$ has d columns.

For $i = 1, \ldots, n$,

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

Theorem (Subspace Embedding from Subsampling)

For each i, and fixed constant c, let $p_i = \min\left(1, \frac{c \log d}{\epsilon^2} \cdot \tau_i\right)$. *Let* $\tilde{\mathbf{A}}$ *have rows sampled from* $\mathbf{A}$ *with probabilities* $p_1, \ldots, p_n$. *With probability* $9/10$,

$$(1-\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2,$$

and $\tilde{\mathbf{A}}$ *has* $O(d \log d/\epsilon^2)$ *rows in expectation.*

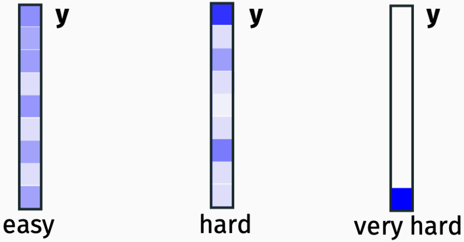
How should we choose the probabilities $p_1, \ldots, p_n$?

As usual, consider a single vector $\mathbf{x}$ and understand how to sample to preserve norm of $\mathbf{y} = \mathbf{A}\mathbf{x}$:

$$\|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 = \|\mathbf{S}\mathbf{A}\mathbf{x}\|_2^2 = \|\mathbf{S}\mathbf{y}\|_2^2 \approx \|\mathbf{y}\|_2^2 = \|\mathbf{A}\mathbf{x}\|_2^2.$$

Then we can union bound over an ϵ-net to extend to all $\mathbf{x}$.

As discussed a few lectures ago, uniform sampling only works well if $\mathbf{y} = \mathbf{A}\mathbf{x}$ is "flat".



Instead consider sampling with probabilities at least proportional to the magnitude of $\mathbf{y}$'s entries:

$$p_i > c \cdot \frac{y_i^2}{\|y\|_2^2} \text{ for constant } c \text{ to be determined.}$$

Let $\tilde{\mathbf{y}}$ be the subsampled $\mathbf{y}$. Recall that, when sampling with probabilities $p_1, \ldots, p_n$, for $i = 1, \ldots, n$ we add y_i to $\tilde{\mathbf{y}}$ with probability p_i and reweight by $\frac{1}{\sqrt{p_i}}$.

$$\|\tilde{\mathbf{y}}\|_2^2 = \sum_{i=1}^{n} \frac{y_i^2}{p_i} \cdot Z_i \quad \text{where} \quad Z_i = \begin{cases} 1 \text{ with probability } p_i \\ 0 \text{ otherwise} \end{cases}$$

$$\mathrm{Var}[\|\tilde{\mathbf{y}}\|_2^2] = \sum_{i=1}^{n} \frac{y_i^2}{p_i} \cdot \mathrm{Var}[Z_i] \leq \sum_{i=1}^{n} \frac{y_i^4}{p_i^2} \cdot p_i = \frac{y_i^4}{p_i}$$

We set $p_i = c \cdot \frac{y_i^2}{\|y\|_2^2}$ so get total variance:

$$\frac{1}{c}\|y\|_2^4$$

Using a Bernstein bound (or Chebyshev's inequality if you don't care about the δ dependence) we have that if $c = \frac{\log(1/\delta)}{\epsilon^2}$ then:

$$\Pr\left[\left|\|\tilde{\mathbf{y}}\|_2^2 - \|\mathbf{y}\|_2^2\right| \geq \epsilon\|\mathbf{y}\|_2^2\right] \leq \delta.$$

The number of samples we take in expectation is:

$$\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} c \cdot \frac{y_i^2}{\|y_i\|_2^2} = \frac{\log(1/\delta)}{\epsilon^2}.$$

We don't know $y_1, \ldots, y_n$! And in fact, these values aren't fixed. We wanted to prove a bound for $\mathbf{y} = \mathbf{A}\mathbf{x}$ for any $\mathbf{x}$.

Idea behind leverage scores: Sample row i from $\mathbf{A}$ using the worst case (largest necessary) sampling probability:

$$\tau_i = \max_{\mathbf{x}} \frac{y_i^2}{\|\mathbf{y}\|_2^2} \qquad \text{where} \qquad \mathbf{y} = \mathbf{A}\mathbf{x}.$$

If we sample with probability $p_i = \frac{1}{\epsilon^2} \cdot \tau_i$, then we will be sampling by at least $\frac{1}{\epsilon^2} \cdot \frac{y_i^2}{\|\mathbf{y}\|_2^2}$, no matter what $\mathbf{y}$ is.

Two concerns:

1) How to compute $\tau_1, \ldots, \tau_n$?

2) the number of samples we take will be roughly $\sum_{i=1}^n \tau_i$. How do we bound this?

$$\tau_i = \max_{\mathbf{x}} \frac{y_i^2}{\|\mathbf{y}\|_2^2} \qquad \text{where} \qquad \mathbf{y} = \mathbf{A}\mathbf{x}.$$

Recall Cauchy-Schwarz inequality: $(\mathbf{w}^T\mathbf{z})^2 \leq \mathbf{w}^T\mathbf{w} \cdot \mathbf{z}^T\mathbf{z}$

Leverage score sampling:

- For $i = 1, \ldots, n$,
  - Compute $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$.
  - Set $p_i = \frac{c \log(1/\delta)}{\epsilon^2} \cdot \tau_i$.
  - Add row $\mathbf{a}_i$ to $\tilde{\mathbf{A}}$ with probability p_i and reweight by $\frac{1}{\sqrt{p_i}}$.

For any fixed $\mathbf{x}$, we will have that
$(1-\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2$ with probability $(1-\delta)$.

How many rows do we sample in expectation?

Claim: No matter how large n is, $\sum_{i=1}^{n} \tau_i = d$ a matrix $\mathbf{A} \in \mathbb{R}^d$.

"Zero-sum" law for the importance of matrix rows.

Leverage score sampling:

- For $i = 1, \ldots, n$,
  - Compute $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$.
  - Set $p_i = \frac{c \log(1/\delta}{\epsilon^2} \cdot \tau_i$.
  - Add row $\mathbf{a}_i$ to $\tilde{\mathbf{A}}$ with probability p_i and reweight by $\frac{1}{\sqrt{p_i}}$.

For any fixed $\mathbf{x}$, we will have that $(1-\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2$ with high probability.

And since $\sum_{i=1}^n p_i = \frac{c \log(1/\delta}{\epsilon^2} \cdot \sum_{i=1}^n \tau_i$, $\tilde{\mathbf{A}}$ contains $O\left(\frac{d \log(1/\delta)}{\epsilon^2}\right)$ rows in expectation.

Last step: need to extend to all $\mathbf{x}$.

Naive ϵ-net argument leads to d^2 dependence since we need to set $\delta = c^d$. Getting the right $d \log d$ dependence below requires a "matrix Chernoff bound" (see e.g. Tropp 2015).

Theorem (Subspace Embedding from Subsampling)

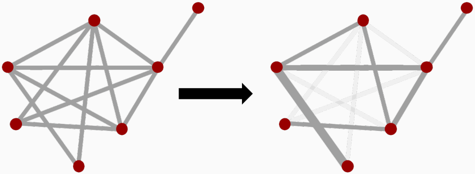
For each i, and fixed constant c, let $p_i = \min\left(1, \frac{c \log d}{\epsilon^2} \cdot \tau_i\right)$. *Let* $\tilde{\mathbf{A}}$ *have rows sampled from* $\mathbf{A}$ *with probabilities* $p_1, \ldots, p_n$. *With probability* $9/10$,

$$(1-\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{A}}\mathbf{x}\|_2^2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}\|_2^2,$$

and $\tilde{\mathbf{A}}$ *has* $O(d \log d/\epsilon^2)$ *rows in expectation.*

SPECTRAL SPARSIFICATION COROLLARY

For any graph G with d nodes, there exists a graph $\tilde{G}$ with $O(d \log d/\epsilon^2)$ edges such that, for all $\mathbf{x}$, $\|\tilde{\mathbf{B}}\mathbf{x}\|_2^2 = (1 \pm \epsilon)\|\mathbf{B}\mathbf{x}\|_2^2$.

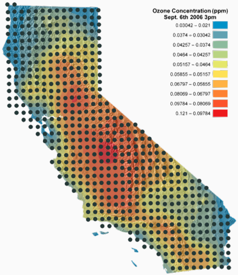


As a result, the value of any cut in $\tilde{G}$ is within a $(1 \pm \epsilon)$ factor of the value in G, the Laplacian eigenvalues are with a $(1 \pm \epsilon)$ factors, etc.

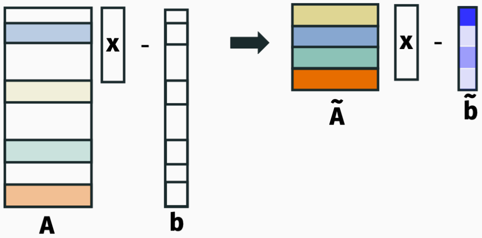
In many applications, computational costs are second order to data collection costs. We have a huge range of possible data points $\mathbf{a}_1, \ldots, \mathbf{a}_n$ that we can collect labels/values $b_1, \ldots, b_n$ for. Goal is to learn $\mathbf{x}$ such that:

$$\mathbf{a}_i^T \mathbf{x} \approx b_i.$$

Want to do so after observing as few $b_1, \ldots, b_n$ as possible. Applications include healthcare, environmental science, etc.



Can be solved via random sampling for linear models.



Claim: Let $\tilde{\mathbf{A}}$ is an $O(1)$-factor subspace embedding for $\mathbf{A}$ (obtained via leverage score sampling). Then $\tilde{\mathbf{x}} = \arg\min \|\tilde{\mathbf{A}}\mathbf{x} - \tilde{\mathbf{b}}\|_2^2$ satisfies:

$$\|\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b}\|_2^2 \leq O(1)\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2^2,$$

where $\mathbf{x}^* = \arg\min \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$. Computing $\tilde{\mathbf{x}}$ only requires collecting $O(d \log d)$ labels (independent of n).

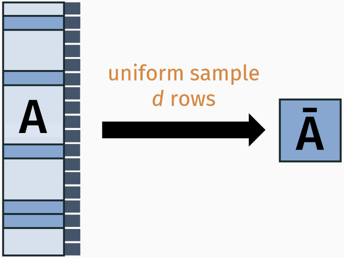
Lots of applications:

- Robust bandlimited and multiband interpolation [STOC 2019].
- Active learning for Gaussian process regression [NeurIPS 2020].
- Active learning beyond the ℓ_2 norm [Preprint 2021]
- Active learning for polynomial regression [Preprint 2021]
- DOE Grant on “learning based” algorithms for solving parametric partial differential equations.

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.

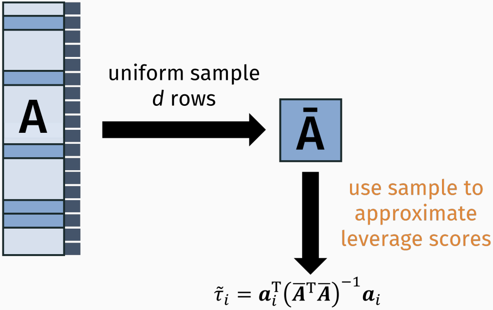
Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.



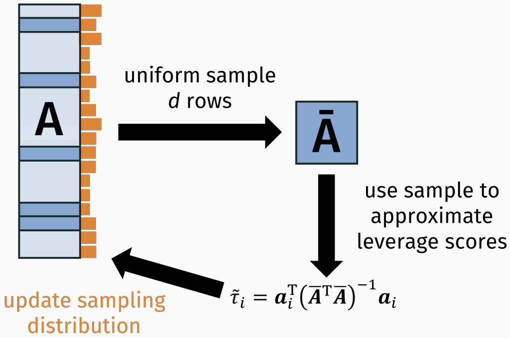
Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.



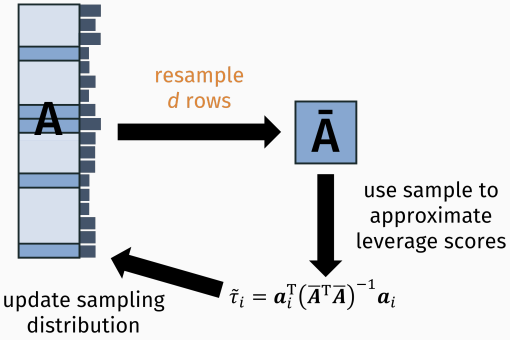
Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.



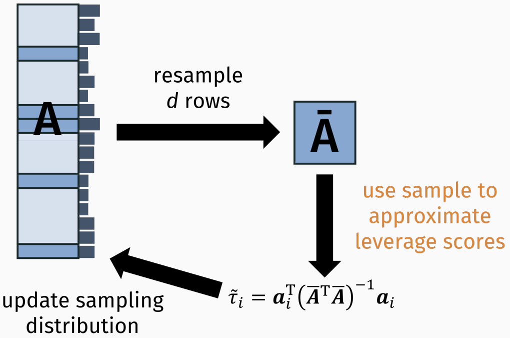
Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.



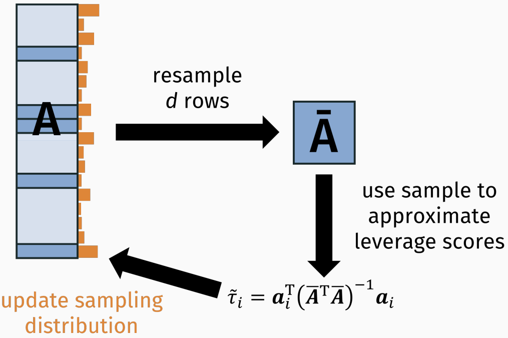
Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.



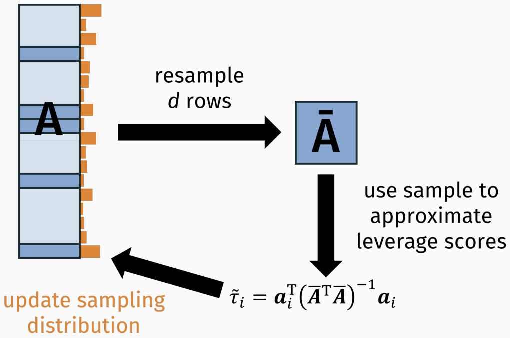
Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.



Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Computing leverage scores $\tau_i = \mathbf{a}_i^T(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{a}_i$ is expensive.



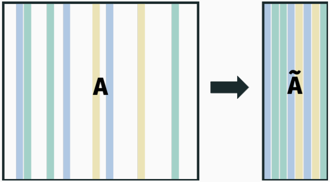
After $O(\log n)$ rounds, $\tilde{\tau}_i \approx \tau_i$ for all i.

Main algorithmic idea: Bootstrap leverage score sampling from uniform sampling (ITCS 2015).

Problem: Sometimes we want to compress down to $\ll d$ rows or columns. E.g. we don't need a full subspace embedding, but just want to find a near optimal rank k approximation.

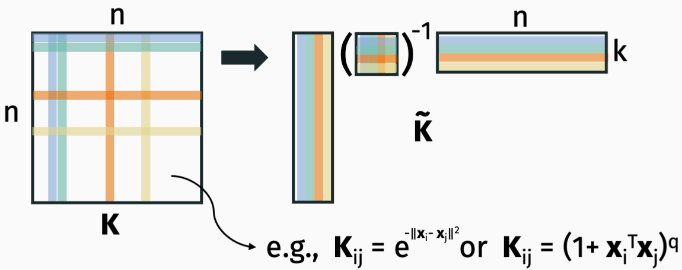
Approach: Use "regularized" version of the leverage scores:

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



Result: Sample $O(k \log k/\epsilon)$ columns whose span contains a near-optimal low-approximation to $\mathbf{A}$ (SODA 2017).

The first $O(nk^2/\epsilon^2)$ time algorithm[1] for near optimal rank-k approximation of any $n \times n$ positive semidefinite kernel matrix:



Based on the classic Nyström method. Importantly, does not even require constructing K explicitly, which takes $O(n^2)$ time.

[1]NeurIPS 2017.

Highlights of the semester for me:

- Very active office hours!
- Large number of students presenting at the reading group. Got to learn about a lot of your reseach interests.
- Lots of collaboration between students.