

# Randomized Numerical Linear Algebra (RandNLA): Past, Present, and Future

---

Petros Drineas<sup>1</sup>

&

Michael W. Mahoney<sup>2</sup>

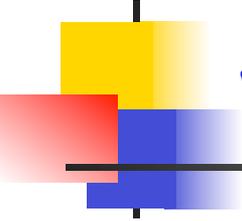
<sup>1</sup> Department of Computer Science, Rensselaer Polytechnic Institute

<sup>2</sup> Department of Mathematics, Stanford University

To access our web pages:

[Google Drineas](#)

[Google Michael Mahoney](#)



# Why RandNLA?

---

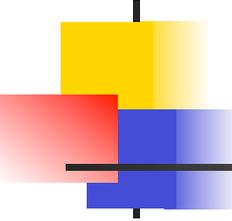
**Randomization and sampling** allow us to design **provably accurate algorithms** for problems that are:

➤ **Massive**

(matrices so large that can not be stored at all, or can only be stored in slow memory devices)

➤ **Computationally expensive** or **NP-hard**

(combinatorial optimization problems such as the Column Subset Selection Problem)



# RandNLA: sampling rows/columns

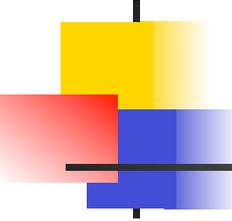
---

## Randomized algorithms

- By (carefully) **sampling rows/columns of a matrix**, we can construct new, smaller matrices that are close to the original matrix (w.r.t. matrix norms) with high probability.

$$\begin{pmatrix} A \end{pmatrix} \cdot \begin{pmatrix} B \end{pmatrix} \approx \begin{pmatrix} C \end{pmatrix} \cdot \begin{pmatrix} R \end{pmatrix}$$

- By **preprocessing the matrix using random projections**, we can sample rows/columns much less carefully (uniformly at random) and still get nice bounds with high probability.



# RandNLA: sampling rows/columns

---

## Randomized algorithms

- By (carefully) **sampling rows/columns of a matrix**, we can construct new, smaller matrices that are close to the original matrix (w.r.t. matrix norms) with high probability.

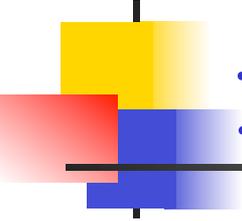
$$\begin{pmatrix} A \end{pmatrix} \cdot \begin{pmatrix} B \end{pmatrix} \approx \begin{pmatrix} C \end{pmatrix} \cdot \begin{pmatrix} R \end{pmatrix}$$

- By **preprocessing the matrix using random projections**, we can sample rows/columns much less carefully (uniformly at random) and still get nice bounds with high probability.

## Matrix perturbation theory

- The resulting smaller matrices behave similarly (in terms of singular values and singular vectors) to the original matrices thanks to the norm bounds.

**Structural results that “decouple” the “randomized” part from the “matrix perturbation” part are important in the analyses of such algorithms.**



# Interplay

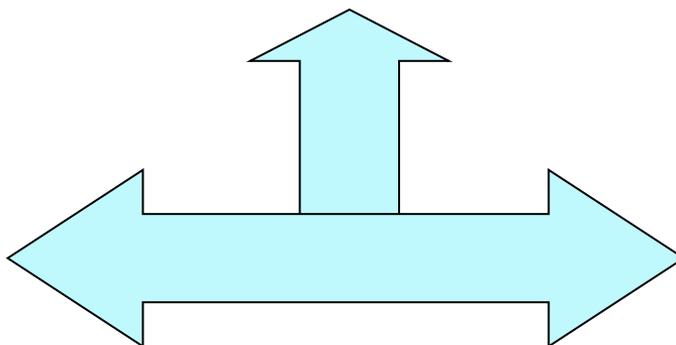
---

## Applications in BIG DATA

(Data Mining, Information Retrieval,  
Machine Learning, Bioinformatics, etc.)

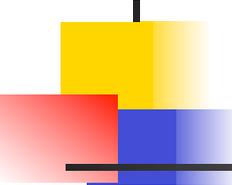
### Theoretical Computer Science

Randomized and approximation  
algorithms



### Numerical Linear Algebra

Matrix computations and linear  
algebra (ie., perturbation theory)



# Roadmap of the tutorial

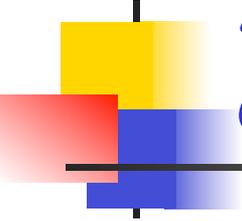
---

**Focus:** sketching matrices (i) by sampling rows/columns and (ii) via “random projections.”

**Machinery:** (i) Approximating matrix multiplication, and (ii) decoupling “randomization” from “matrix perturbation.”

## **Overview of the tutorial:**

- (i) Motivation: computational efficiency, interpretability
- (ii) Approximating matrix multiplication
- (iii) From matrix multiplication to  $CX/CUR$  factorizations and approximate SVD
- (iv) Improvements and recent progress
- (v) Algorithmic approaches to least-squares problems
- (vi) Statistical perspectives on least-squares algorithms
- (vii) Theory and practice of: extending these ideas to kernels and SPSD matrices
- (viii) Theory and practice of: implementing these ideas in large-scale settings



# Areas of RandNLA that will not be covered in this tutorial

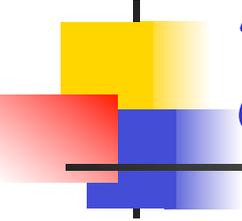
---

## Element-wise sampling

- Introduced by Achlioptas and McSherry in *STOC* 2001.
- **Current state-of-the-art**: additive error bounds for arbitrary matrices and exact reconstruction under (very) restrictive assumptions  
(important breakthroughs by Candes, Recht, Tao, Wainright, and others)
- **To do**: Efficient, optimal, relative-error accuracy algorithms for arbitrary matrices.

## Solving systems of linear equations

- Almost optimal relative-error approximation algorithms for Laplacian and Symmetric Diagonally Dominant (SDD) matrices (Spielman, Teng, Miller, Koutis, and others).
- **To do**: progress beyond Laplacians.



# Areas of RandNLA that will not be covered in this tutorial

---

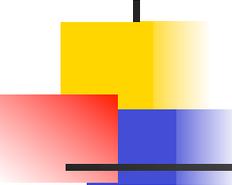
## Element-wise sampling

- Introduced by Achlioptas and McSherry in *STOC* 2001.
- **Current state-of-the-art**: additive error bounds for arbitrary matrices and exact reconstruction under (very) restrictive assumptions  
(important breakthroughs by Candes, Recht, Tao, Wainright, and others)
- **To do**: Efficient, optimal, relative-error accuracy algorithms for arbitrary matrices.

## Solving systems of linear equations

- Almost optimal relative-error approximation algorithms for Laplacian and Symmetric Diagonally Dominant (SDD) matrices (Spielman, Teng, Miller, Koutis, and others).
- **To do**: progress beyond Laplacians.

**There are surprising (?) connections between all three areas (row/column sampling, element-wise sampling, and solving systems of linear equations).**



# Roadmap of the tutorial

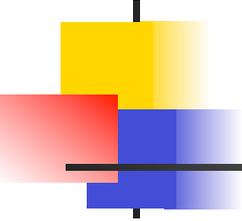
---

**Focus:** sketching matrices (i) by sampling rows/columns and (ii) via “random projections.”

**Machinery:** (i) Approximating matrix multiplication, and (ii) decoupling “randomization” from “matrix perturbation.”

## **Overview of the tutorial:**

- (i) Motivation: computational efficiency, interpretability
- (ii) Approximating matrix multiplication
- (iii) From matrix multiplication to  $CX/CUR$  factorizations and approximate SVD
- (iv) Improvements and recent progress
- (v) Algorithmic approaches to least-squares problems
- (vi) Statistical perspectives on least-squares algorithms
- (vii) Theory and practice of: extending these ideas to kernels and SPSD matrices
- (viii) Theory and practice of: implementing these ideas in large-scale settings



# Human genetics

---

**Single Nucleotide Polymorphisms:** the most common type of genetic variation in the genome across different individuals.

They are **known** locations at the human genome where **two** alternate nucleotide bases (**alleles**) are observed (out of A, C, G, T).

SNPs



Matrices including thousands of individuals and hundreds of thousands if SNPs are available.

## HGDP data

- 1,033 samples
- 7 geographic regions
- 52 populations

## The Human Genome Diversity Panel (HGDP)

### Africans

- 1 Bantu
- 2 Mandenka
- 3 Yoruba
- 4 San
- 5 Mbuti pygmy
- 6 Biaka
- 7 Mozabite

Cavalli-Sforza (2005) *Nat Genet Rev*

Rosenberg et al. (2002) *Science*

Li et al. (2008) *Science*

### Europeans

- 8 Orcadian
- 9 Adygei
- 10 Russian
- 11 Basque
- 12 French
- 13 North Italian
- 14 Sardinian
- 15 Tuscan

### Western Asians

- 16 Bedouin
- 17 Druze
- 18 Palestinian

### Central and Southern Asians

- 19 Balochi
- 20 Brahui
- 21 Makrani
- 22 Sindhi
- 23 Pathan
- 24 Burusho
- 25 Hazara
- 26 Uygur
- 27 Kalash

### Eastern Asians

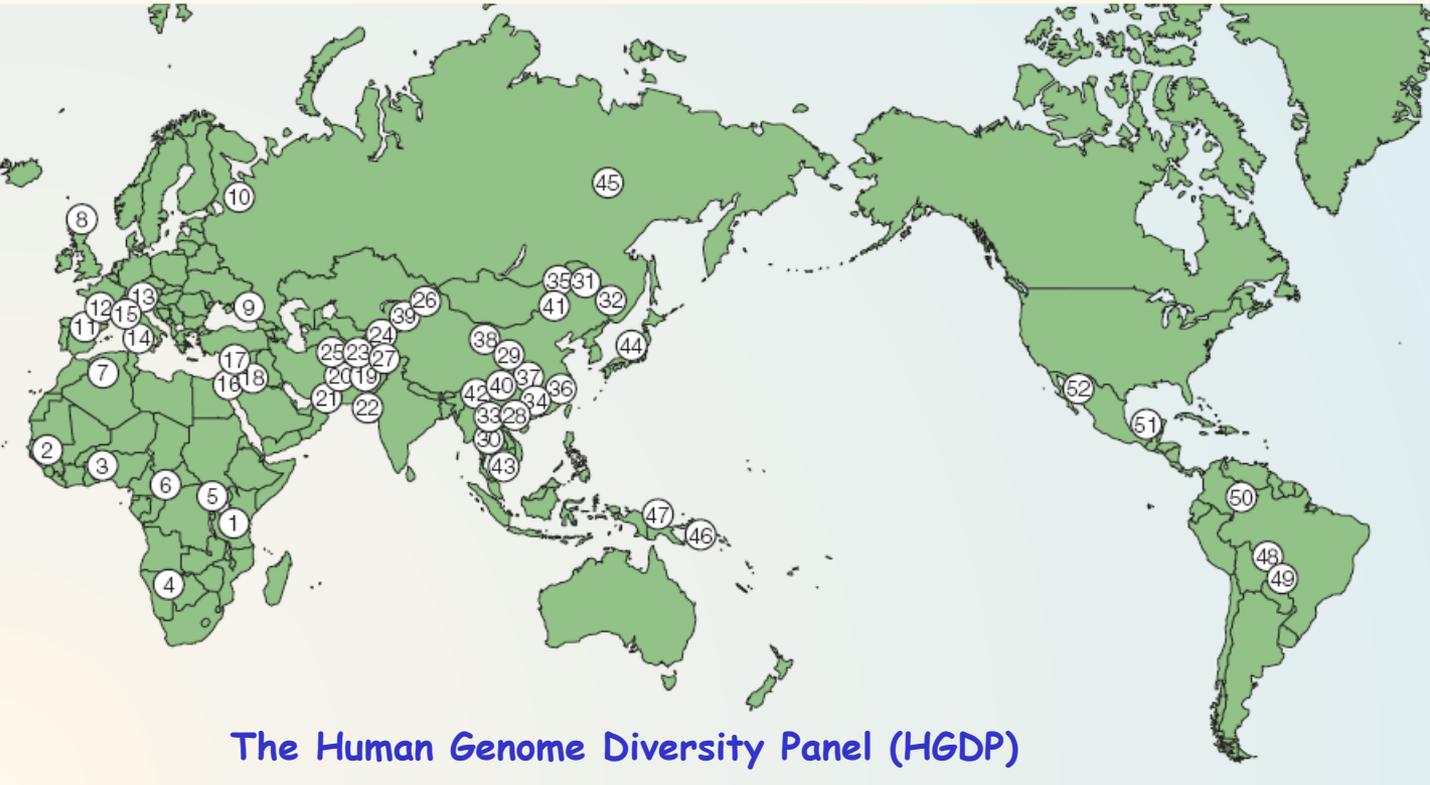
- 28 Han (S. China)
- 29 Han (N. China)
- 30 Dai
- 31 Daur
- 32 Hezhen
- 33 Lahu
- 34 Miao
- 35 Oroqen
- 36 She
- 37 Tujia
- 38 Tu
- 39 Xibo
- 40 Yi
- 41 Mongola
- 42 Naxi
- 43 Cambodian
- 44 Japanese
- 45 Yakut

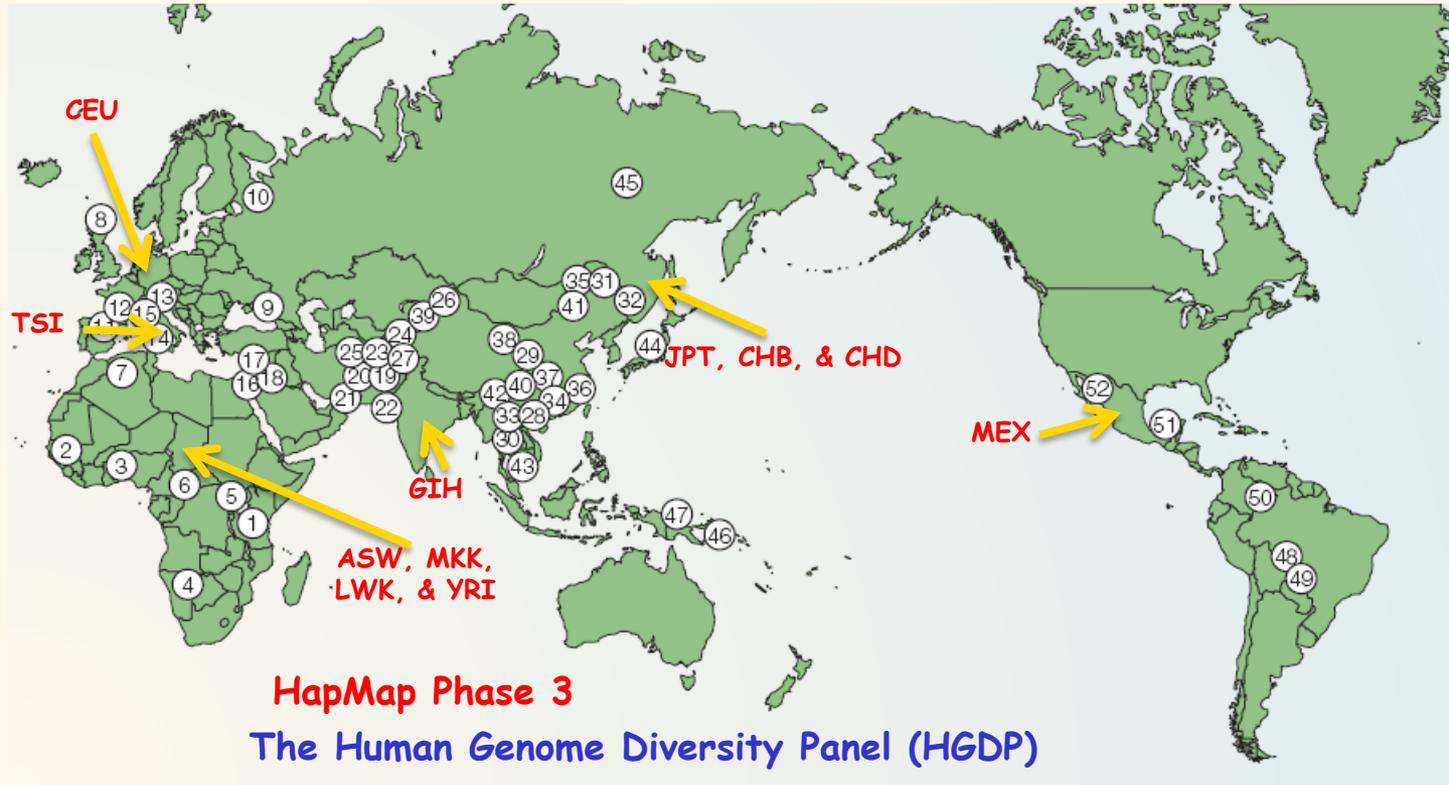
### Oceanians

- 46 Melanesian
- 47 Papuan

### Native Americans

- 48 Karitiana
- 49 Surui
- 50 Colombian
- 51 Maya
- 52 Pima





### HGDP data

- 1,033 samples
- 7 geographic regions
- 52 populations

### HapMap Phase 3 data

- 1,207 samples
- 11 populations

#### Africans

- 1 Bantu
- 2 Mandenka
- 3 Yoruba
- 4 San
- 5 Mbuti pygmy
- 6 Biaka
- 7 Mozabite

#### Europeans

- 8 Orcadian
- 9 Adygei
- 10 Russian
- 11 Basque
- 12 French
- 13 North Italian
- 14 Sardinian
- 15 Tuscan

#### Western Asians

- 16 Bedouin
- 17 Druze
- 18 Palestinian

#### Central and Southern Asians

- 19 Balochi
- 20 Brahui
- 21 Makrani
- 22 Sindhi
- 23 Pathan
- 24 Burusho
- 25 Hazara
- 26 Uygur
- 27 Kalash

#### Eastern Asians

- 28 Han (S. China)
- 29 Han (N. China)
- 30 Dai
- 31 Daur
- 32 Hezhen
- 33 Lahu
- 34 Miao
- 35 Oroqen
- 36 She
- 37 Tujia
- 38 Tu
- 39 Xibo
- 40 Yi
- 41 Mongola
- 42 Naxi
- 43 Cambodian
- 44 Japanese
- 45 Yakut

#### Oceanians

- 46 Melanesian
- 47 Papuan

#### Native Americans

- 48 Karitiana
- 49 Surui
- 50 Colombian
- 51 Maya
- 52 Pima

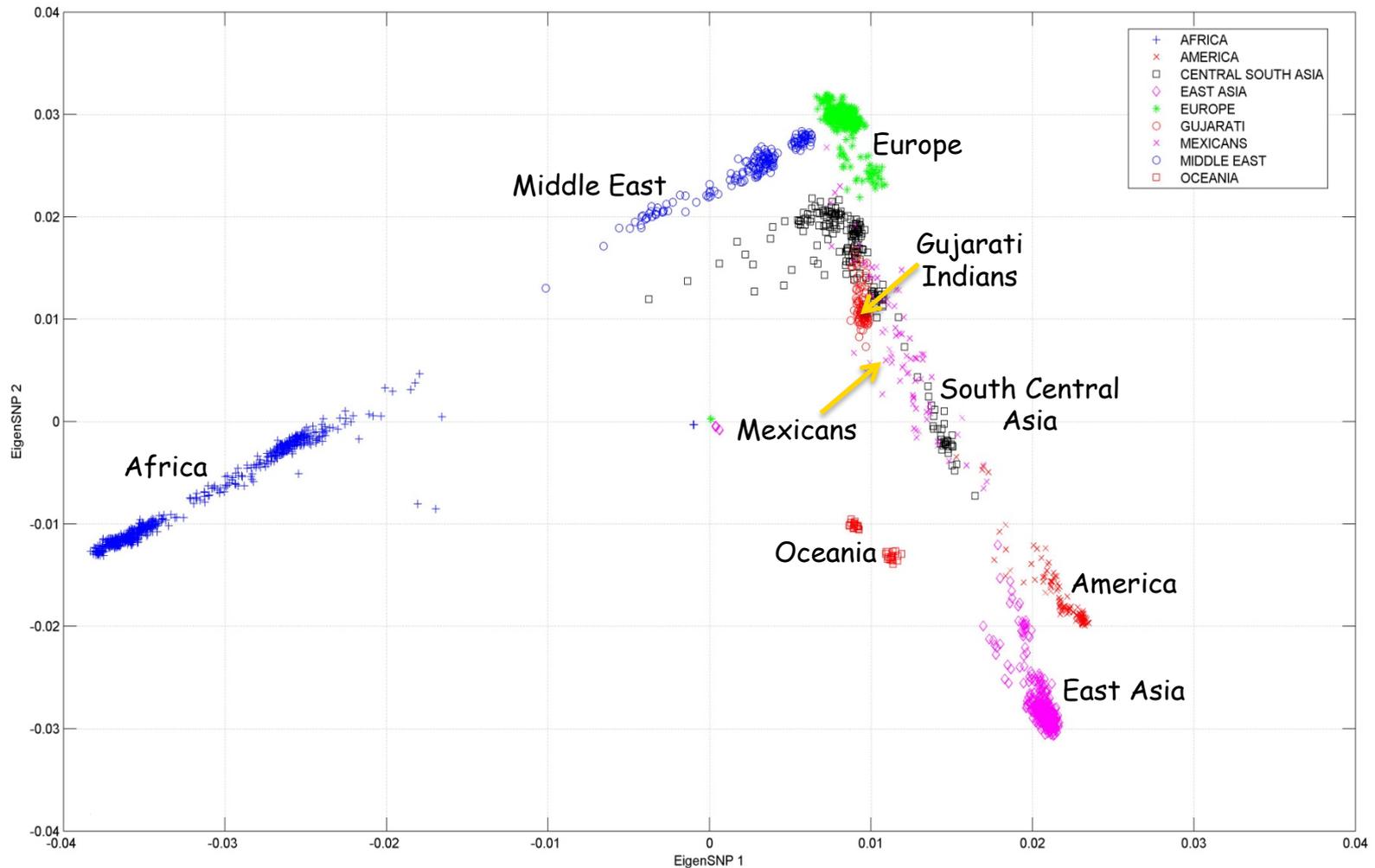
Cavalli-Sforza (2005) *Nat Genet Rev*

Rosenberg et al. (2002) *Science*

Li et al. (2008) *Science*

The International HapMap Consortium  
 (2003, 2005, 2007) *Nature*

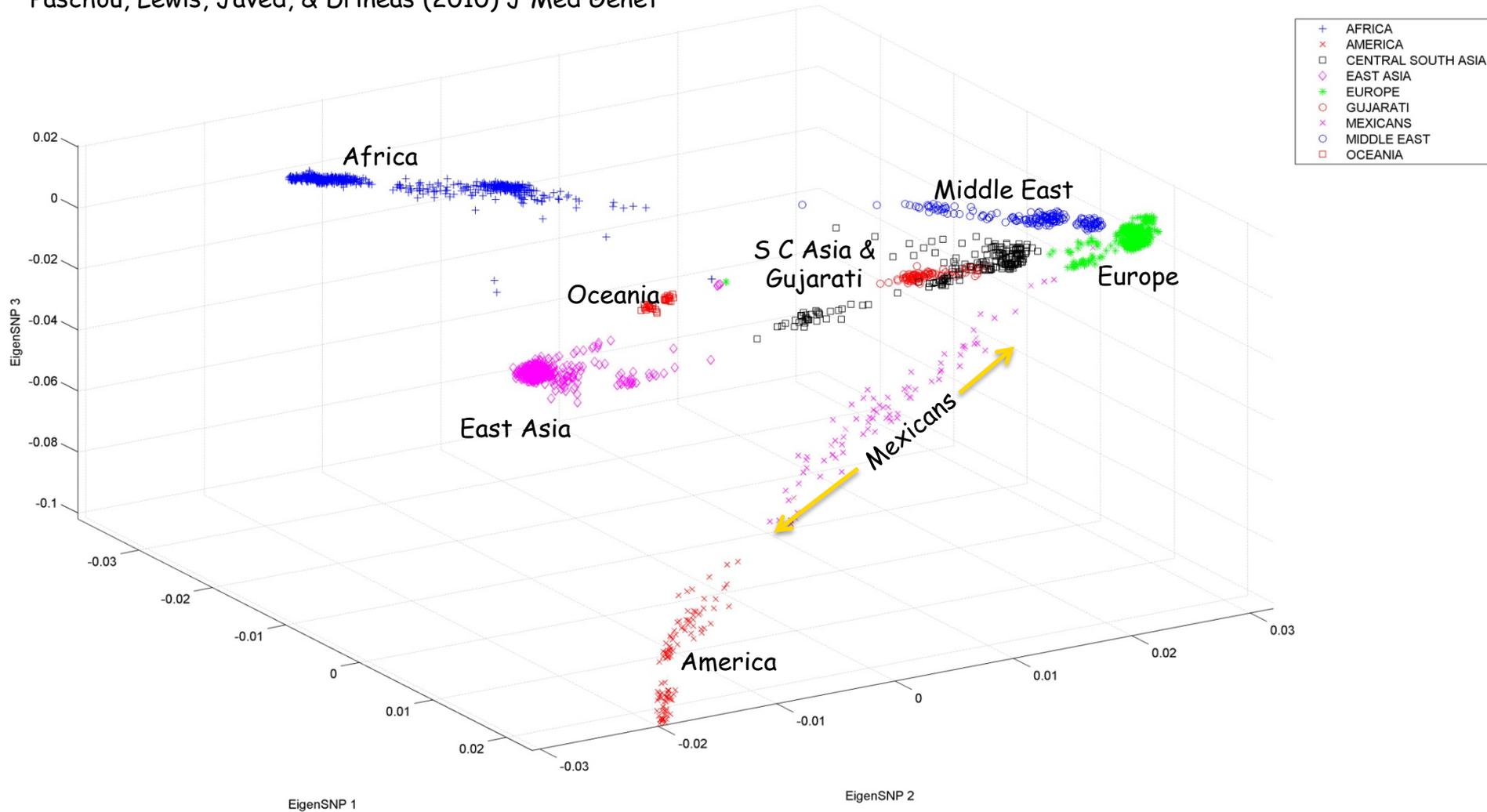




- Top two Principal Components (PCs or eigenSNPs)

(Lin and Altman (2005) *Am J Hum Genet*)

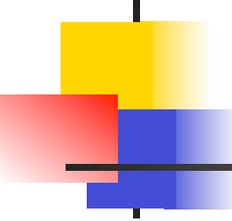
- The figure renders visual support to the "out-of-Africa" hypothesis.
- Mexican population seems out of place: we move to the top three PCs.



**Not altogether satisfactory:** the principal components are linear combinations of all SNPs, and - of course - can not be assayed!

Can we find **actual SNPs** that capture the information in the singular vectors?

Formally: **spanning the same subspace.**



# Issues

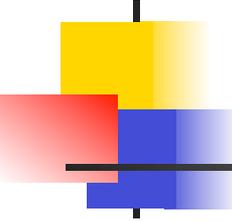
---

- **Computing large SVDs: computational time**

- **In commodity hardware** (e.g., a 4GB RAM, dual-core laptop), using MatLab 7.0 (R14), the computation of the SVD of the dense 2,240-by-447,143 matrix  $A$  takes about 12 minutes.
- Computing this SVD is not a one-liner, since we can not load the whole matrix in RAM (runs out-of-memory in MatLab).
- We compute the eigendecomposition of  $AA^T$ .
- In a similar experiment, we had to compute the SVD of a **14,267-by-14,267 matrix to analyze mitochondrial DNA from 14,267 samples from approx. 120 populations** in order to infer the relatedness of the **Ancient Minoans to Modern European, Northern African, and Middle Eastern populations**.

(Hughey, Paschou, Drineas, et. al. (2013) Nature Communications)

- **Obviously, running time is a concern.**
- **Machine-precision accuracy is NOT necessary!**
  - Data are noisy.
  - Approximate singular vectors work well in our settings.



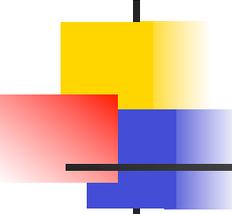
## Issues (cont'd)

---

- **Selecting good columns that “capture the structure” of the top PCs**
  - Combinatorial optimization problem; hard even for small matrices.
  - Often called the Column Subset Selection Problem (CSSP).
  - Not clear that such columns even exist.

### **The two issues:**

- (i) Fast approximation to the top  $k$  singular vectors of a matrix, and
- (ii) Selecting columns that capture the structure of the top  $k$  singular vectors **are connected and can be tackled using the same framework.**



# SVD decomposes a matrix as...

---

$$\begin{pmatrix} m \times n \\ A \end{pmatrix} \approx \begin{pmatrix} m \times k \\ U_k \end{pmatrix} \begin{pmatrix} k \times n \\ X \end{pmatrix}$$

↑  
Top  $k$  left singular vectors

The SVD has strong optimality properties.

- It is easy to see that  $X = U_k^T A$ .
- SVD has strong optimality properties.
- The columns of  $U_k$  are linear combinations of up to all columns of  $A$ .

# The CX decomposition

Mahoney & Drineas (2009) PNAS

$$\begin{pmatrix} m \times n \\ A \end{pmatrix} \approx \begin{pmatrix} m \times c \\ C \end{pmatrix} \begin{pmatrix} c \times n \\ X \end{pmatrix}$$

Carefully chosen X

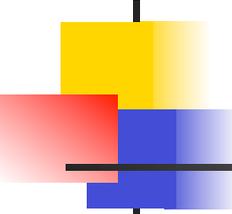
Goal: make (some norm) of  $A-CX$  small.

$c$  columns of  $A$ , with  $c$  being as close to  $k$  as possible

## Why?

If  $A$  is an subject-SNP matrix, then selecting representative columns is equivalent to selecting representative SNPs to capture the same structure as the top eigenSNPs.

We want  $c$  as small as possible!



# CX decomposition

---

$$\begin{pmatrix} m \times n \\ A \end{pmatrix} \approx \begin{pmatrix} m \times c \\ C \end{pmatrix} \begin{pmatrix} c \times n \\ X \end{pmatrix}$$

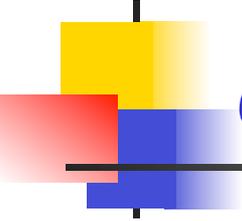
$c$  columns of  $A$ , with  $c$  being  
as close to  $k$  as possible

Easy to prove that optimal  $X = C^+A$ .

(with respect to unitarily invariant norms;  $C^+$  is the Moore-Penrose pseudoinverse of  $C$ ).

Thus, the challenging part is to find **good columns (SNPs) of  $A$  to include in  $C$** .

From a mathematical perspective, this is a combinatorial optimization problem, closely related to the so-called **Column Subset Selection Problem (CSSP)**; the CSSP has been heavily studied in Numerical Linear Algebra.



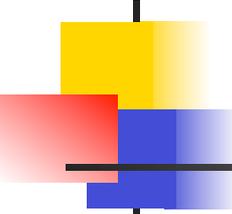
# Our objective for the $CX$ decomposition

---

We would like to get theorems of the following form:

Given an  $m$ -by- $n$  matrix  $A$ , there exists an *efficient* algorithm that picks a *small* number of columns of  $A$  such that with *reasonable* probability:

$$\|A - CX\|_F = \left\| A - CC^\dagger A \right\|_F \leq (1 + \varepsilon) \|A - A_k\|_F$$

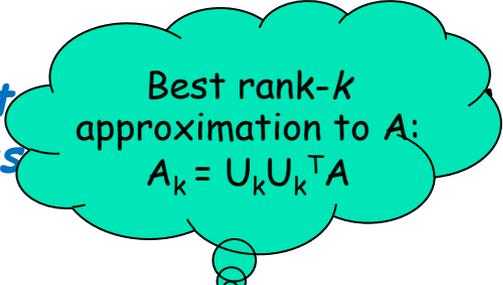


# Our objective for the $CX$ decomposition

---

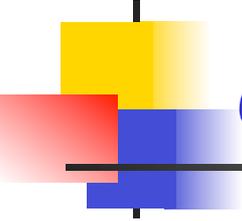
We would like to get theorems of the following form:

Given an  $m$ -by- $n$  matrix  $A$ , there exists an *efficient* way to find a *small* number of columns of  $A$  such that with *reasonable*



Best rank- $k$  approximation to  $A$ :  
 $A_k = U_k U_k^T A$

$$\|A - CX\|_F = \|A - CC^\dagger A\|_F \leq (1 + \varepsilon) \|A - A_k\|_F$$



# Our objective for the CX decomposition

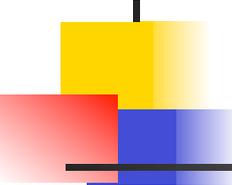
---

We would like to get theorems of the following form:

low-degree polynomial  
in  $m$ ,  $n$ , and  $k$

Given an  $m$ -by- $n$  matrix  $A$ , there exists an **efficient** algorithm that picks a **small** number of columns of  $A$  such that with **reasonable** probability:  
Close to  $k/\varepsilon$  constant, high, almost surely, etc.

$$\|A - CX\|_F = \left\| A - CC^\dagger A \right\|_F \leq (1 + \varepsilon) \|A - A_k\|_F$$



# Roadmap of the tutorial

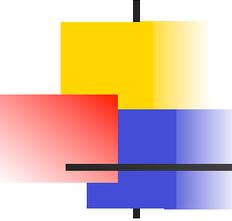
---

**Focus:** sketching matrices (i) by sampling rows/columns and (ii) via “random projections.”

**Machinery:** (i) Approximating matrix multiplication, and (ii) decoupling “randomization” from “matrix perturbation.”

## **Overview of the tutorial:**

- (i) Motivation: computational efficiency, interpretability
- (ii) Approximating matrix multiplication
- (iii) From matrix multiplication to  $CX/CUR$  factorizations and approximate SVD
- (iv) Improvements and recent progress
- (v) Algorithmic approaches to least-squares problems
- (vi) Statistical perspectives on least-squares algorithms
- (vii) Theory and practice of: extending these ideas to kernels and SPSD matrices
- (viii) Theory and practice of: implementing these ideas in large-scale settings



# Approximating Matrix Multiplication

---

## Problem Statement

Given an  $m$ -by- $n$  matrix  $A$  and an  $n$ -by- $p$  matrix  $B$ , approximate the product  $A \cdot B$ ,

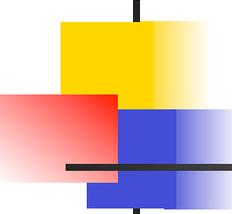
**OR, equivalently,**

Approximate the  $\text{sum of } n \text{ rank-one matrices.}$

$$A \cdot B = \sum_{i=1}^n \underbrace{\begin{pmatrix} \mathbf{A}^{(i)} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{B}^{(i)} \end{pmatrix}}_{\in \mathbb{R}^{m \times p}}$$

$\mathbf{A}^{(i)}$  is the  $i$ -th column of  $A$ .  
 $\mathbf{B}^{(i)}$  is the  $i$ -th row of  $B$ .

Each term in the summation is a rank-one matrix



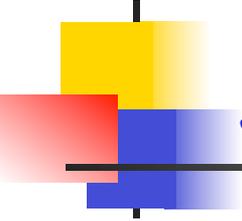
# A sampling approach

---

$$A \cdot B = \sum_{i=1}^n \underbrace{\begin{pmatrix} \mathbf{A}^{(i)} \end{pmatrix}}_{\substack{\text{i-th column of } A \\ \in \mathbb{R}^{m \times p}}} \cdot \begin{pmatrix} \mathbf{B}^{(i)} \end{pmatrix}_{\substack{\text{i-th row of } B}}$$

## Algorithm

1. Fix a set of probabilities  $p_i$ ,  $i=1\dots n$ , summing up to 1.
2. For  $t=1\dots c$ ,  
set  $j_t = i$ , where  $P(j_t = i) = p_i$ .  
(Pick  $c$  terms of the sum, with replacement, with respect to the  $p_i$ .)
3. Approximate the product  $AB$  by summing the  $c$  terms, after scaling.



# Sampling (cont'd)

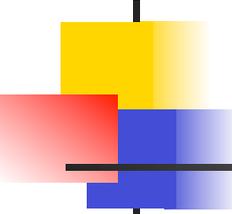
---

$$A \cdot B = \sum_{i=1}^n \underbrace{\begin{pmatrix} \mathbf{A}^{(i)} \end{pmatrix}}_{\in \mathbb{R}^{m \times p}} \cdot \begin{pmatrix} \mathbf{B}^{(i)} \end{pmatrix}$$

$\mathbf{A}^{(i)}$  is the  $i$ -th column of  $A$ .  
 $\mathbf{B}^{(i)}$  is the  $i$ -th row of  $B$ .

$$\approx \frac{1}{c} \sum_{t=1}^c \frac{1}{p_{j_t}} \underbrace{\begin{pmatrix} \mathbf{A}^{(j_t)} \end{pmatrix}}_{\in \mathbb{R}^{m \times p}} \cdot \begin{pmatrix} \mathbf{B}^{(j_t)} \end{pmatrix}$$

Keeping the terms  $j_1, j_2, \dots, j_c$ .



# The algorithm (matrix notation)

---

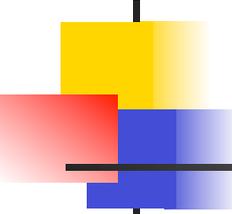
$$\begin{pmatrix} A \\ m \times n \end{pmatrix} \cdot \begin{pmatrix} B \\ n \times p \end{pmatrix} \approx \begin{pmatrix} C \\ m \times c \end{pmatrix} \cdot \begin{pmatrix} R \\ c \times p \end{pmatrix}$$

## Algorithm

1. Pick  $c$  columns of  $A$  to form an  $m$ -by- $c$  matrix  $C$  and the corresponding  $c$  rows of  $B$  to form a  $c$ -by- $p$  matrix  $R$ .
2. Approximate  $A \cdot B$  by  $C \cdot R$ .

## Notes

3. We pick the columns and rows with non-uniform probabilities.
4. We scale the columns (rows) prior to including them in  $C$  ( $R$ ).



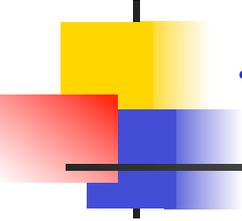
## The algorithm (matrix notation, cont'd)

$$\begin{pmatrix} A \\ m \times n \end{pmatrix} \cdot \begin{pmatrix} B \\ n \times p \end{pmatrix} \approx \begin{pmatrix} C \\ m \times c \end{pmatrix} \cdot \begin{pmatrix} R \\ c \times p \end{pmatrix}$$

- Create  $C$  and  $R$  by performing  $c$  i.i.d. trials, with replacement.
- For  $t = 1 \dots c$ , pick a column  $A^{(j_t)}$  and a row  $B_{(j_t)}$  with probability

$$\mathbb{P}(j_t = i) = \frac{\|A^{(i)}\|_2 \|B_{(i)}\|_2}{\sum_{j=1}^n \|A^{(j)}\|_2 \|B_{(j)}\|_2}$$

- Include  $A^{(j_t)}/(cp_{j_t})^{1/2}$  as a column of  $C$ , and  $B_{(j_t)}/(sp_{j_t})^{1/2}$  as a row of  $R$ .



## The algorithm (matrix notation, cont'd)

---

We can also use the sampling matrix notation:

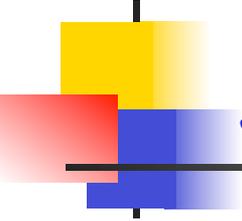
Let  $S$  be an  $n$ -by- $c$  matrix whose  $t$ -th column (for  $t = 1 \dots c$ ) has a single non-zero entry, namely

$$S_{jtt} = \frac{1}{\sqrt{cp_{jt}}}$$

Clearly:

$$A \cdot B \approx C \cdot R = (AS) \cdot (S^T B)$$

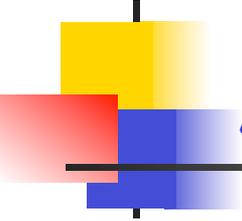
Note:  $S$  is sparse (has exactly  $c$  non-zero elements, one per column).



# Simple Lemmas

---

- It is easy to implement this particular sampling in two passes.
- The **expectation of CR (element-wise) is AB** (unbiased estimator), regardless of the sampling probabilities.
- Our particular choice of sampling probabilities **minimizes the variance** of the estimator (w.r.t. the Frobenius norm of the error  $AB-CR$ ).



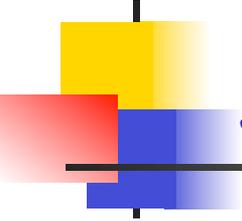
## A bound for the Frobenius norm

---

For the above algorithm,

$$\mathbb{E} [\|AB - CR\|_F] = \mathbb{E} [\|AB - ASS^T B\|_F] \leq \frac{1}{\sqrt{c}} \|A\|_F \|B\|_F$$

- This is *easy to prove* (elementary manipulations of expectation).
- Measure concentration follows from a *martingale argument*.
- The above bound also implies an upper bound for the spectral norm of the error  $AB - CR$ .



## Special case: $B = A^T$

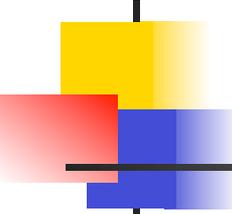
---

If  $B = A^T$ , then the sampling probabilities are

$$\mathbb{P}(j_t = i) = \frac{\|A^{(i)}\|_2^2}{\|A\|_F^2}$$

Also,  $R = C^T$ , and the error bounds are:

$$\mathbb{E} [\|AA^T - CC^T\|_F] = \mathbb{E} [\|AA^T - ASS^T A^T\|_F] \leq \frac{1}{\sqrt{c}} \|A\|_F^2$$



# Special case: $B = A^T$ (cont'd)

(Drineas et al. Num Math 2011, Theorem 4)

---

A better **spectral norm** bound via matrix Chernoff/Bernstein inequalities:

## Assumptions:

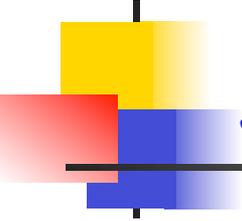
- Spectral norm of  $A$  is one (not important, just normalization)
- Frobenius norm of  $A$  is at least 0.2 (not important, simplifies bounds).

- **Important:** Set

$$c = \Omega \left( \frac{\|A\|_F^2}{\epsilon^2} \ln \left( \frac{\|A\|_F^2}{\epsilon^2 \sqrt{\delta}} \right) \right)$$

Then: for any  $0 < \epsilon < 1$ , with probability at least  $1 - \delta$ ,

$$\|AA^T - CC^T\|_2 = \|AA^T - ASS^T A^T\|_2 \leq \epsilon$$

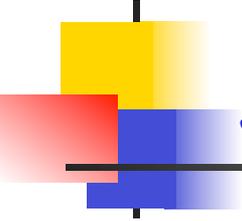


## Special case: $B = A^T$ (cont'd)

---

### Notes:

- The constants hidden in the big-Omega notation are small.
- The proof is simple: an immediate application of an inequality derived by Oliveira (2010) for sums of random Hermitian matrices.
- Similar results were first proven by Rudelson & Vershynin (2007) JACM, but the proofs were much more complicated.
- We need a sufficiently large value of  $c$ , otherwise the theorem **does not work**.



## Special case: $B = A^T$ (cont'd)

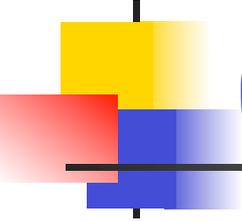
---

### Notes:

- The constants hidden in the big-Omega notation are small.
- The proof is simple: an immediate application of an inequality derived by Oliveira (2010) for sums of random Hermitian matrices.
- Similar results were first proven by Rudelson & Vershynin (2007) JACM, but the proofs were much more complicated.
- We need a sufficiently large value of  $c$ , otherwise the theorem **does not work**.

### Open problems:

- Non-trivial upper bounds for other unitarily invariant norms.  
E.g., Schatten  $p$ -norms for other values of  $p$ . Especially for  $p = 1$  (trace norm).
- Upper bounds for non-unitarily invariant norms that might be useful in practice.



## Using a dense $S$ (instead of a sampling matrix...)

---

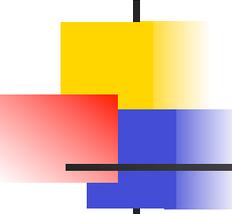
We approximated the product  $AB$  as follows:

$$A \cdot B \approx C \cdot R = (AS) \cdot (S^T B)$$

Recall that  $S$  is an  $n$ -by- $c$  sparse matrix (one non-zero entry per column).

Let's replace  $S$  by a dense matrix, the random sign matrix:

$$S_{ij} = \begin{cases} +1/\sqrt{c} & ,\text{w.p. } 1/2 \\ -1/\sqrt{c} & ,\text{w.p. } 1/2 \end{cases}$$



## Using a dense $S$ (instead of a sampling matrix...)

---

We approximated the product  $AB$  as follows:

$$A \cdot B \approx C \cdot R = (AS) \cdot (S^T B)$$

Recall that  $S$  is an  $n$ -by- $c$  sparse matrix (one non-zero entry per column).

Let's replace  $S$  by a dense matrix, the random sign matrix:

$$S_{ij} = \begin{cases} +1/\sqrt{c} & , \text{w.p. } 1/2 \\ -1/\sqrt{c} & , \text{w.p. } 1/2 \end{cases}$$

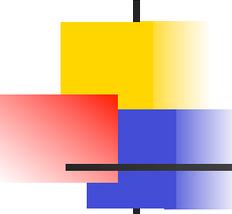
$\text{st}(A)$ : stable rank of  $A$   
 $\text{st}(A) = \|A\|_F^2 / \|A\|_2^2$

If

$$c = \Omega \left( \max \{ \text{st}(A), \text{st}(B) \} \ln(m+p) / \epsilon^2 \right)$$

then, with high probability (see Theorem 3.1 in Magen & Zouzias SODA 2012)

$$\|AB - CR\|_2 = \|AB - ASS^T B\|_2 \leq \epsilon \|A\|_2 \|B\|_2$$



# Using a dense $S$ (instead of a sampling matrix...) (and focusing on $B = A^T$ , normalized)

---

Approximate the product  $AA^T$  (assuming that the spectral norm of  $A$  is one):

$$A \cdot A^T \approx C \cdot C^T = (AS) \cdot (S^T A^T)$$

Let  $S$  be a dense matrix, the random sign matrix:

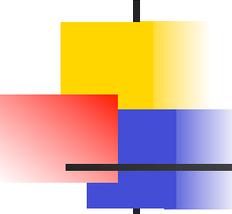
$$S_{ij} = \begin{cases} +1/\sqrt{c} & , \text{w.p. } 1/2 \\ -1/\sqrt{c} & , \text{w.p. } 1/2 \end{cases}$$

If

$$c = \Omega \left( \frac{\|A\|_F^2}{\epsilon^2} \ln m \right)$$

then, with high probability:

$$\|AA^T - CC^T\|_2 = \|AA^T - ASS^T A^T\|_2 \leq \epsilon$$



# Using a dense $S$ (instead of a sampling matrix...) (and focusing on $B = A^T$ , normalized)

---

Approximate the product  $AA^T$  (assuming that the spectral norm of  $A$  is one):

$$A \cdot A^T \approx C \cdot C^T = (AS) \cdot (S^T A^T)$$

Let  $S$  be a dense matrix, the random sign matrix:

$$S_{ij} = \begin{cases} +1/\sqrt{c} & , \text{w.p. } 1/2 \\ -1/\sqrt{c} & , \text{w.p. } 1/2 \end{cases}$$

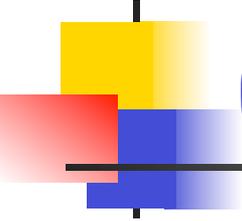
If

$$c = \Omega \left( \frac{\|A\|_F^2}{\epsilon^2} \ln m \right)$$

Similar structure with the sparse  $S$  case; some differences in the  $\ln$  factor

then, with high probability:

$$\|AA^T - CC^T\|_2 = \|AA^T - ASS^T A^T\|_2 \leq \epsilon$$

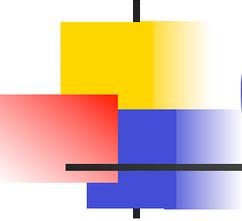


## Using a dense $S$ (cont'd)

---

### Comments:

- This matrix multiplication approximation is oblivious to the input matrices  $A$  and  $B$ .
- Reminiscent of random projections and the Johnson-Lindenstrauss (JL) transform.
- Bounds for the Frobenius norm are easier to prove and are very similar to the case where  $S$  is just a sampling matrix.
- We need a sufficiently large value for  $c$ , otherwise the (spectral norm) theorem does not hold.
- It holds for arbitrary  $A$  and  $B$  (not just  $B = A^T$ ); the sampling-based approach should also be generalizable to arbitrary  $A$  and  $B$ .



## Using a dense $S$ (cont'd)

---

### Other choices for dense matrices $S$ ?

#### Why bother with a sign matrix?

(Computing the product  $AS$  and  $S^T B$  is somewhat slow, taking  $O(mnc)$  and  $O(pnc)$  time.)

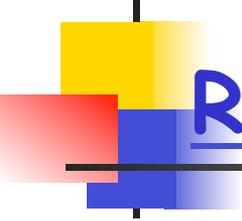
**Similar** bounds are known for better, i.e., **computationally more efficient**, choices of “random projection” matrices  $S$ , most notably:

- When  $S$  is the so-called **subsampled Hadamard Transform Matrix**.

(much faster; avoids full matrix-matrix multiplication; see Sarlos FOCS 2006 and Drineas et al. (2011) Num Math)

- When  $S$  is the **ultra-sparse projection matrix of Clarkson & Woodruff STOC 2013**.

(the matrix multiplication result appears in Mahoney & Meng STOC 2013).



# Recap: approximating matrix multiplication

We approximated the product  $AB$  as follows:

$$A \cdot B \approx C \cdot R = (AS) \cdot (S^T B)$$

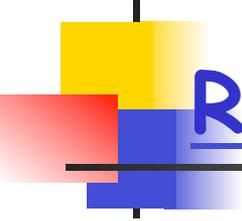
Let  $S$  be a sampling matrix (actual columns from  $A$  and rows from  $B$  are selected):

We need to carefully sample columns of  $A$  (rows of  $B$ ) with probabilities that depend on their norms in order to get "good" bounds of the following form:

$$\mathbb{E} [\|AB - CR\|_F] = \mathbb{E} [\|AB - ASS^T B\|_F] \leq \frac{1}{\sqrt{c}} \|A\|_F \|B\|_F$$

$$\|AA^T - CC^T\|_2 = \|AA^T - ASS^T A^T\|_2 \leq \varepsilon$$

↑  
Holds with probability at least  $1-\delta$  by setting  $c = \Omega\left(\frac{\|A\|_F^2}{\varepsilon^2} \ln\left(\frac{\|A\|_F^2}{\varepsilon^2 \sqrt{\delta}}\right)\right)$



# Recap: approximating matrix multiplication

Alternatively, we approximated the product  $AB$  as follows:

$$A \cdot B \approx C \cdot R = (AS) \cdot (S^T B)$$

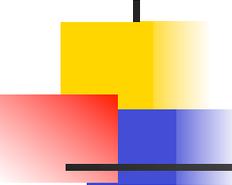
Now  $S$  is a random projection matrix (linear combinations of columns of  $A$  and rows of  $B$  are formed).

Oblivious to the actual input matrices  $A$  and  $B$ !

$$\mathbb{E} [\|AB - CR\|_F] = \mathbb{E} [\|AB - ASS^T B\|_F] \leq \frac{1}{\sqrt{c}} \|A\|_F \|B\|_F$$

$$\|AA^T - CC^T\|_2 = \|AA^T - ASS^T A^T\|_2 \leq \epsilon$$

Holds with high probability by setting  $c = \Omega\left(\frac{\|A\|_F^2}{\epsilon^2} \ln m\right)$   
(for the random sign matrix)



# Roadmap of the tutorial

---

**Focus:** sketching matrices (i) by sampling rows/columns and (ii) via “random projections.”

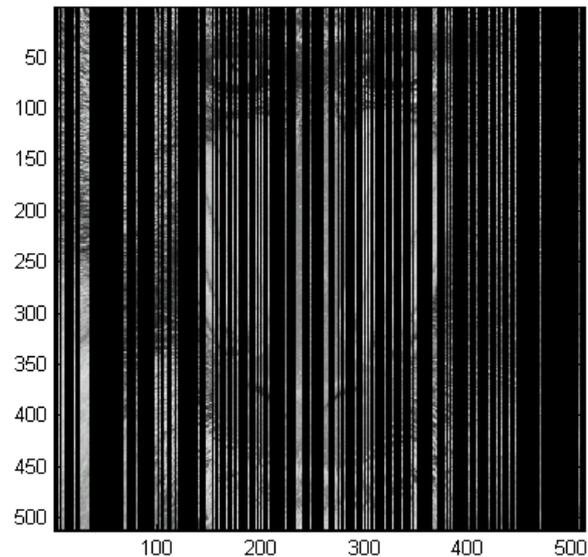
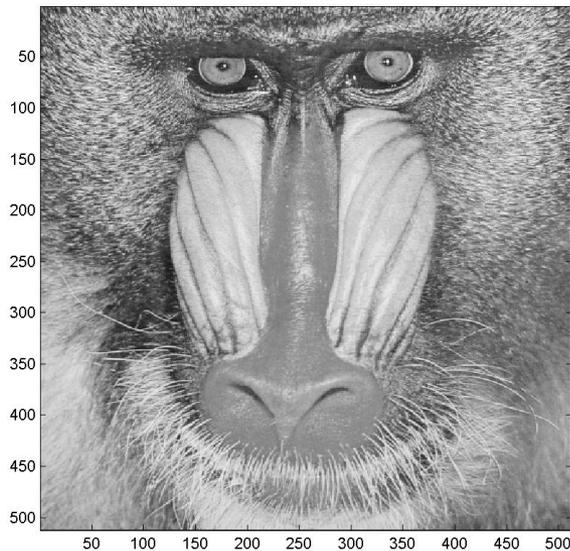
**Machinery:** (i) Approximating matrix multiplication, and (ii) decoupling “randomization” from “matrix perturbation.”

## **Overview of the tutorial:**

- (i) Motivation: computational efficiency, interpretability
- (ii) Approximating matrix multiplication
- (iii) From matrix multiplication to  $CX/CUR$  factorizations and approximate SVD
- (iv) Improvements and recent progress
- (v) Algorithmic approaches to least-squares problems
- (vi) Statistical perspectives on least-squares algorithms
- (vii) Theory and practice of: extending these ideas to kernels and SPSD matrices
- (viii) Theory and practice of: implementing these ideas in large-scale settings



# Approximating singular vectors



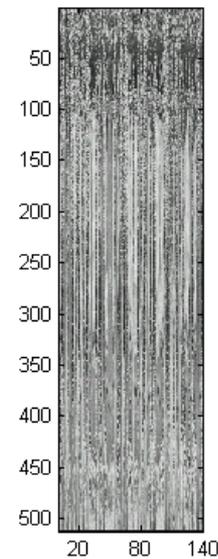
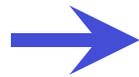
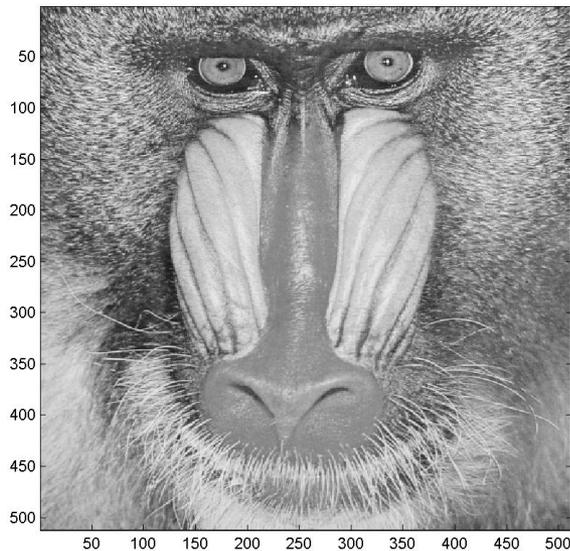
Original matrix

Sampling ( $c = 140$  columns)

1. Sample  $c (=140)$  columns of the original matrix  $A$  and rescale them appropriately to form a  $512$ -by- $c$  matrix  $C$ .
2. Show that  $A-CX$  is "small".

( $C^+$  is the pseudoinverse of  $C$  and  $X= C^+A$ )

# Approximating singular vectors



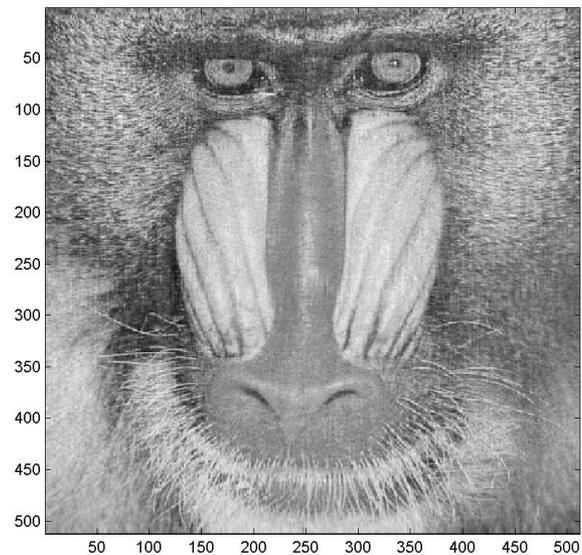
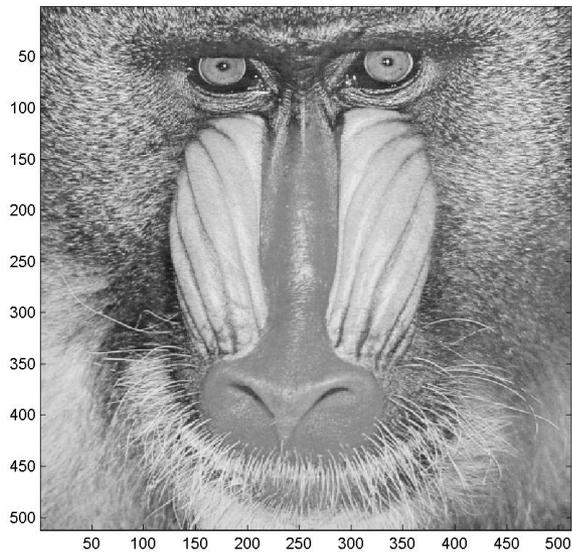
Original matrix

Sampling ( $c = 140$  columns)

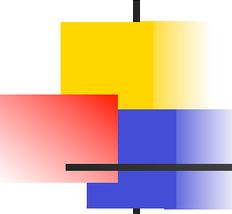
1. Sample  $c (=140)$  columns of the original matrix  $A$  and rescale them appropriately to form a  $512$ -by- $c$  matrix  $C$ .
2. Show that  $A-CX$  is "small".

( $C^+$  is the pseudoinverse of  $C$  and  $X= C^+A$ )

# Approximating singular vectors (cont'd)



The fact that  $AA^T - CC^T$  is small will imply that  $A-CX$  is small as well.



## Proof (spectral norm)

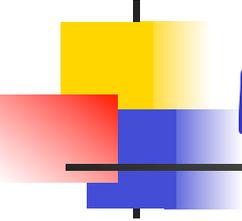
---

Using the triangle inequality and properties of norms,

$$\begin{aligned}\|A - CC^\dagger A\|_2^2 &= \|(I - CC^\dagger)A\|_2^2 \\ &= \|(I - CC^\dagger)AA^T(I - CC^\dagger)^T\|_2 \\ &= \|(I - CC^\dagger)(AA^T - CC^T)(I - CC^\dagger)^T\|_2 \\ &\leq \|AA^T - CC^T\|_2\end{aligned}$$

projector matrices

We used the fact that  $(I - CC^\dagger)CC^T$  is equal to zero.



## Proof (spectral norm), cont'd

---

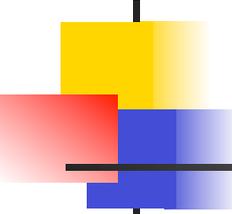
Assume that our sampling is done in  $c$  i.i.d. trials and the sampling probabilities are:

$$\mathbb{P}(j_t = i) = \frac{\|A^{(i)}\|_2^2}{\|A\|_F^2}$$

We can use our matrix multiplication result:

(We will upper bound the spectral norm by the Frobenius norm to avoid concerns about  $c$ , namely whether  $c$  exceeds the threshold necessitated by the theory.)

$$\begin{aligned} \mathbb{E} \left[ \left\| A - CC^\dagger A \right\|_2 \right] &\leq \mathbb{E} \left[ \left\| AA^T - CC^T \right\|_2 \right] \\ &\leq \frac{1}{c^{1/4}} \|A\|_F \end{aligned}$$



# Is this a good bound?

---

$$\begin{aligned}\mathbb{E} \left[ \left\| A - CC^\dagger A \right\|_2 \right] &\leq \mathbb{E} \left[ \left\| AA^T - CC^T \right\|_2 \right] \\ &\leq \frac{1}{c^{1/4}} \|A\|_F\end{aligned}$$

**Problem 1:** If  $c = n$  we do not get zero error.

That's because of sampling with replacement.

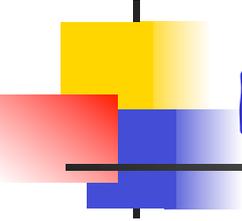
(We know how to analyze uniform sampling without replacement, but we have no bounds on non-uniform sampling without replacement.)

**Problem 2:** If  $A$  had rank exactly  $k$ , we would like a column selection procedure that drives the error down to zero when  $c = k$ .

This can be done deterministically simply by selecting  $k$  linearly independent columns.

**Problem 3:** If  $A$  had numerical rank  $k$ , we would like a bound that depends on the norm of  $A - A_k$  and not on the norm of  $A$ .

Such deterministic bounds exist when  $c = k$  and depend on  $(k(n - k))^{1/2} \|A - A_k\|_2$



# Relative-error Frobenius norm bounds

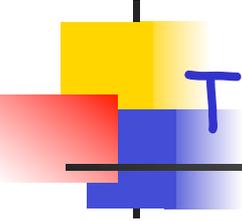
---

Given an  $m$ -by- $n$  matrix  $A$ , there exists an  $O(mn^2)$  algorithm that picks

$O((k/\varepsilon^2) \ln(k/\varepsilon^2))$  columns of  $A$

such that with probability at least 0.9

$$\|A - CX\|_F = \|A - CC^\dagger A\|_F \leq (1 + \varepsilon) \|A - A_k\|_F$$



# The algorithm

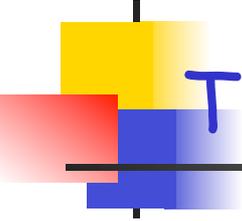
---

Input:  $m$ -by- $n$  matrix  $A$ ,  
 $0 < \epsilon < .5$ , the desired accuracy

Output:  $C$ , the matrix consisting of the selected columns

## Sampling algorithm

- Compute probabilities  $p_j$  summing to 1
- Let  $c = O((k/\epsilon^2) \ln(k/\epsilon^2))$ .
- In  $c$  i.i.d. trials pick columns of  $A$ , where in each trial the  $j$ -th column of  $A$  is picked with probability  $p_j$ .
- Let  $C$  be the matrix consisting of the chosen columns.



# The algorithm

---

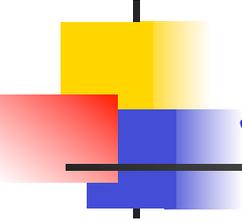
Input:  $m$ -by- $n$  matrix  $A$ ,  
 $0 < \epsilon < .5$ , the desired accuracy

Output:  $C$ , the matrix consisting of the selected columns

## Sampling algorithm

- Compute probabilities  $p_j$  summing to 1
- Let  $c = O((k/\epsilon^2) \ln(k/\epsilon^2))$ .
- In  $c$  i.i.d. trials pick columns of  $A$ , where in each trial the  $j$ -th column of  $A$  is picked with probability  $p_j$ .
- Let  $C$  be the matrix consisting of the chosen columns.

**Note:** there is no rescaling of the columns of  $C$  in this algorithm; however, since our error matrix is  $A - CX = A - CC^+A$ , rescaling the columns of  $C$  (as we did in our matrix multiplication algorithms), does not change  $A - CX = A - CC^+A$ .



# Subspace sampling (Frobenius norm)

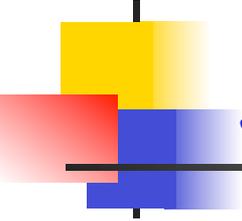
---

$$\begin{pmatrix} A_k \\ m \times n \end{pmatrix} = \begin{pmatrix} U_k \\ m \times k \end{pmatrix} \cdot \begin{pmatrix} \Sigma_k \\ k \times k \end{pmatrix} \cdot \begin{pmatrix} V_k^T \\ k \times n \end{pmatrix}$$

$V_k$ : orthogonal matrix containing the top  $k$  right singular vectors of  $A$ .

$\Sigma_k$ : diagonal matrix containing the top  $k$  singular values of  $A$ .

**Remark:** The rows of  $V_k^T$  are orthonormal vectors, but its columns  $(V_k^T)^{(i)}$  are not.



# Subspace sampling (Frobenius norm)

$$\begin{pmatrix} A_k \\ m \times n \end{pmatrix} = \begin{pmatrix} U_k \\ m \times k \end{pmatrix} \cdot \begin{pmatrix} \Sigma_k \\ k \times k \end{pmatrix} \cdot \begin{pmatrix} V_k^T \\ k \times n \end{pmatrix}$$

$V_k$ : orthogonal matrix containing the top  $k$  right singular vectors of  $A$ .

$\Sigma_k$ : diagonal matrix containing the top  $k$  singular values of  $A$ .

**Remark:** The rows of  $V_k^T$  are orthonormal vectors, but its columns  $(V_k^T)^{(i)}$  are not.

Subspace sampling in  $O(mn^2)$  time

$$p_j = \frac{\left\| (V_k^T)^{(j)} \right\|_2^2}{k}$$

Normalization s.t. the  $p_j$  sum up to 1

# Subspace sampling (Frobenius norm)

$$\begin{pmatrix} A_k \\ m \times n \end{pmatrix} = \begin{pmatrix} U_k \\ m \times k \end{pmatrix} \cdot \begin{pmatrix} \Sigma_k \\ k \times k \end{pmatrix} \cdot \begin{pmatrix} V_k^T \\ k \times n \end{pmatrix}$$

$V_k$ : orthogonal matrix containing the top  $k$  right singular vectors of  $A$ .

$\Sigma_k$ : diagonal matrix containing the top  $k$  singular values of  $A$ .

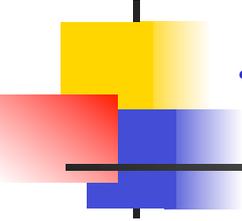
**Remark:** The rows of  $V_k^T$  are orthonormal vectors, but its columns  $(V_k^T)^{(i)}$  are not.

Subspace sampling in  $O(mn^2)$  time

Leverage scores  
(many references in the  
statistics community)

$$p_j = \frac{\left\| (V_k^T)^{(j)} \right\|_2^2}{k}$$

Normalization s.t. the  
 $p_j$  sum up to 1



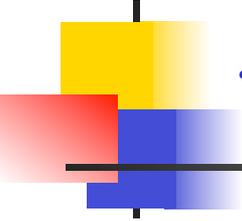
# Towards a relative error bound...

---

Structural result (deterministic):

$$\|A - CX\|_F = \|A - CC^\dagger A\|_F \leq \|A - A_k\|_F + \|(A - A_k)S(V_k^T S)^\dagger\|_F$$

This holds for any  $n$ -by- $c$  matrix  $S$  such that  $C = AS$  as long as the  $k$ -by- $c$  matrix  $V_k^T S$  has full rank (equal to  $k$ ).



# Towards a relative error bound...

---

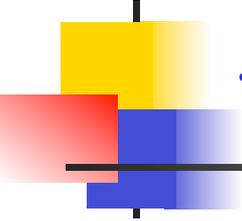
**Structural result (deterministic):**

$$\|A - CX\|_F = \left\| A - CC^\dagger A \right\|_F \leq \|A - A_k\|_F + \left\| (A - A_k) S (V_k^T S)^\dagger \right\|_F$$

This holds for any  $n$ -by- $c$  matrix  $S$  such that  $C = AS$  **as long as the  $k$ -by- $c$  matrix  $V_k^T S$  has full rank (equal to  $k$ ).**

- The proof of the structural result critically uses the fact that with  $X = C^\dagger A$  is the *argmin* for any unitarily invariant norm of the error  $A - CX$ .
- Variants of this structural result have appeared in various papers.

( e.g., (i) Drineas, Mahoney, Muthukrishnan (2008) SIMAX, (ii) Boutsidis, Drineas, Mahoney SODA 2011, (iii) Halko, Martinsson, Tropp (2011) SIREV, (iv) Boutsidis, Drineas, Magdon-Ismail FOCS 2011, etc.)



# The rank of $V_k^T S$

---

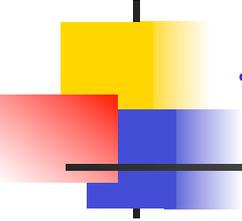
**Structural result (deterministic):**

$$\|A - CX\|_F = \|A - CC^\dagger A\|_F \leq \|A - A_k\|_F + \|(A - A_k) S (V_k^T S)^\dagger\|_F$$

This holds for any  $n$ -by- $c$  matrix  $S$  such that  $C = AS$  as long as the  $k$ -by- $c$  matrix  $V_k^T S$  has full rank (equal to  $k$ ).

$S$  be a sampling and rescaling matrix, where the sampling probabilities are the leverage scores: our matrix multiplication results (and the fact that the square of the Frobenius norm of  $V_k$  is equal to  $k$ ) guarantee that, for our choice of  $c$  (with constant probability):

$$\|V_k^T V_k - V_k^T S S^T V_k\|_2 = \|I_k - V_k^T S S^T V_k\|_2 \leq \varepsilon$$



## The rank of $V_k^T S$ (cont'd)

---

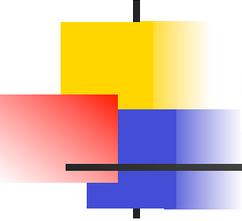
From matrix perturbation theory, if

$$\|V_k^T V_k - V_k^T S S^T V_k\|_2 = \|I_k - V_k^T S S^T V_k\|_2 \leq \varepsilon$$

it follows that all singular values ( $\sigma_i$ ) of  $V_k^T S$  satisfy:

$$\sqrt{1 - \varepsilon} \leq \sigma_i (V_k^T S) \leq \sqrt{1 + \varepsilon}$$

By choosing  $\varepsilon$  small enough, we can guarantee that  $V_k^T S$  has full rank (with constant probability).



## Bounding the second term

---

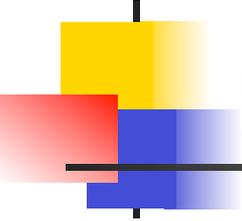
Structural result (deterministic):

$$\|A - CX\|_F = \|A - CC^\dagger A\|_F \leq \|A - A_k\|_F + \|(A - A_k) S (V_k^T S)^\dagger\|_F$$

This holds for any  $n$ -by- $c$  matrix  $S$  such that  $C = AS$  as long as the  $k$ -by- $c$  matrix  $V_k^T S$  has full rank (equal to  $k$ ).

Using strong submultiplicativity for the second term:

$$\begin{aligned} \|(A - A_k) S (V_k^T S)^\dagger\|_F &\leq \|(A - A_k) S\|_F \|(V_k^T S)^\dagger\|_2 \\ &= \sigma_{\min}^{-1}(V_k^T S) \|(A - A_k) S\|_F \end{aligned}$$



## Bounding the second term (cont'd)

---

To conclude:

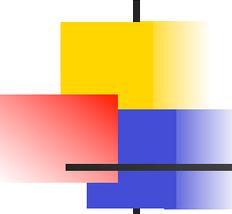
$$\begin{aligned} \left\| (A - A_k) S (V_k^T S)^\dagger \right\|_F &\leq \| (A - A_k) S \|_F \left\| (V_k^T S)^\dagger \right\|_2 \\ &= \sigma_{\min}^{-1} (V_k^T S) \| (A - A_k) S \|_F \end{aligned}$$

(i) We already have a bound for all singular values of  $V_k^T S$  (go back two slides).

(ii) It is easy to prove that, using our sampling and rescaling,

$$\mathbb{E} \left[ \| (A - A_k) S \|_F^2 \right] = \| A - A_k \|_F^2$$

Collecting, we get a  $(2+\epsilon)$  constant-factor approximation.



## Bounding the second term (cont'd)

---

To conclude:

$$\begin{aligned}\left\| (A - A_k) S (V_k^T S)^\dagger \right\|_F &\leq \| (A - A_k) S \|_F \left\| (V_k^T S)^\dagger \right\|_2 \\ &= \sigma_{\min}^{-1} (V_k^T S) \| (A - A_k) S \|_F\end{aligned}$$

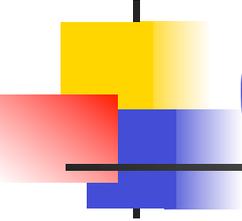
(i) We already have a bound for all singular values of  $V_k^T S$  (go back two slides).

(ii) It is easy to prove that, using our sampling and rescaling,

$$\mathbb{E} \left[ \| (A - A_k) S \|_F^2 \right] = \| A - A_k \|_F^2$$

Collecting, we get a  $(2+\varepsilon)$  constant-factor approximation.

A more careful (albeit, longer) analysis can improve the result to a  $(1+\varepsilon)$  relative-error approximation.



## Using a dense matrix $S$

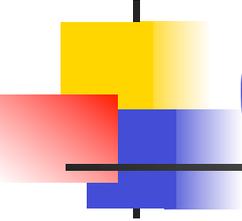
---

Our proof would also work if instead of the sampling matrix  $S$ , we used, for example, the **dense random sign matrix**  $S$ :

$$S_{ij} = \begin{cases} +1/\sqrt{c} & ,\text{w.p. } 1/2 \\ -1/\sqrt{c} & ,\text{w.p. } 1/2 \end{cases}$$

**The intuition is clear:** the most critical part of the proof is based on approximate matrix multiplication to bound the singular values of  $V_k^T S$ .

This also works when  $S$  is a dense matrix.



# Using a dense matrix $S$

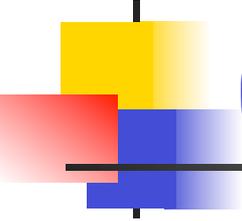
---

## Notes:

**Negative:**  $C=AS$  does not consist of columns of  $A$  (interpretability is lost).

**Positive:** It can be shown that the span of  $C=AS$  contains "relative-error" approximations to the top  $k$  left singular vectors of  $A$ , which can be computed in  $O(nc^2)$  time.

Thus, we can compute approximations to the top  $k$  left singular vectors of  $A$  in  $O(mnc + nc^2)$  time, **already faster than** the naïve  $O(\min\{mn^2, m^2n\})$  time of the **full SVD**.



# Using a dense matrix $S$

---

## Notes:

**Negative:**  $C=AS$  does not consist of columns of  $A$  (interpretability is lost).

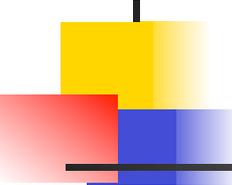
**Positive:** It can be shown that the span of  $C=AS$  contains "relative-error" approximations to the top  $k$  left singular vectors of  $A$ , which can be computed in  $O(nc^2)$  time.

Thus, we can compute approximations to the top  $k$  left singular vectors of  $A$  in  $O(mnc + nc^2)$  time, **already faster than** the naïve  $O(\min\{mn^2, m^2n\})$  time of the **full SVD**.

**Even better:** Using very fast random projections (the **Fast Hadamard Transform**, or the **Clarkson-Woodruff sparse projection**), we can reduce the (first term of the) running time further to

$$O(mn \text{ polylog}(n)).$$

Implementations are simple and work very well in practice!



# Roadmap of the tutorial

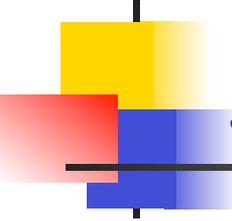
---

**Focus:** sketching matrices (i) by sampling rows/columns and (ii) via “random projections.”

**Machinery:** (i) Approximating matrix multiplication, and (ii) decoupling “randomization” from “matrix perturbation.”

## **Overview of the tutorial:**

- (i) Motivation: computational efficiency, interpretability
- (ii) Approximating matrix multiplication
- (iii) From matrix multiplication to  $CX/CUR$  factorizations and approximate SVD
- (iv) Improvements and recent progress
- (v) Algorithmic approaches to least-squares problems
- (vi) Statistical perspectives on least-squares algorithms
- (vii) Theory and practice of: extending these ideas to kernels and SPSD matrices
- (viii) Theory and practice of: implementing these ideas in large-scale settings



# Selecting fewer columns

---

## Problem

How many columns do we need to include in the matrix  $C$  in order to get relative-error approximations?

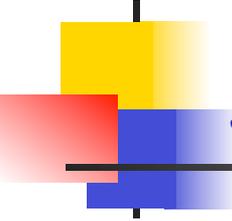
**Recall:** with  $O(k/\epsilon^2 \log(k/\epsilon^2))$  columns, we get (subject to a failure probability)

$$\left\| A - CC^\dagger A \right\|_F \leq (1 + \epsilon) \|A - A_k\|_F$$

**Deshpande & Rademacher (FOCS '10):** with exactly  $k$  columns, we get

$$\left\| A - CC^\dagger A \right\|_F \leq \sqrt{k} \|A - A_k\|_F$$

What about the range between  $k$  and  $O(k \log k)$ ?



# Selecting fewer columns (cont'd)

---

(Boutsidis, Drineas, & Magdon-Ismail, FOCS 2011)

## Question:

What about the range between  $k$  and  $O(k \log k)$ ?

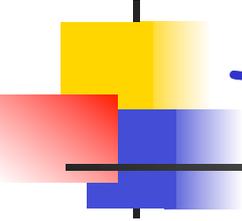
## Answer:

A relative-error bound is possible by selecting  $c=3k/\epsilon$  columns!

## Technical breakthrough:

A combination of sampling strategies with a novel approach on column selection, inspired by the work of Batson, Spielman, & Srivastava (STOC '09) on graph sparsifiers.

- The running time is  $O((mnk+nk^3)\epsilon^{-1})$ .
- Simplicity is gone...



# Towards such a result

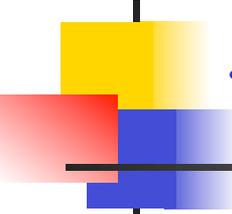
---

First, let the top- $k$  right singular vectors of  $A$  be  $V_k$ .

A structural result (deterministic):

$$\|A - CX\|_F = \|A - CC^\dagger A\|_F \leq \|A - A_k\|_F + \|(A - A_k)S(V_k^T S)^\dagger\|_F$$

Again, this holds for any  $n$ -by- $c$  matrix  $S$  assuming that **the matrix  $V_k^T S$  has full rank** (equal to  $k$ )



## Towards such a result (cont'd)

---

First, let the top- $k$  right singular vectors of  $A$  be  $V_k$ .

A structural result (deterministic):

$$\|A - CX\|_F = \|A - CC^\dagger A\|_F \leq \|A - A_k\|_F + \|(A - A_k)S(V_k^T S)^\dagger\|_F$$

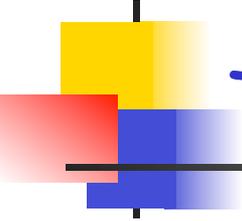
Again, this holds for any  $n$ -by- $c$  matrix  $S$  assuming that **the matrix  $V_k^T S$  has full rank** (equal to  $k$ )

We would like to get a sampling and rescaling matrix  $S$  such that, simultaneously,

$$\sigma_{\min}(V_k^T S) \geq 1 - \sqrt{\frac{k}{c}} \quad \text{and} \quad \|(A - A_k)S\|_F \leq c_0 \|A - A_k\|_F$$

(for some small, fixed constant  $c_0$ ; actually  $c_0 = 1$  in our final result).

Setting  $c = O(k/\varepsilon)$ , we get a  $(2+\varepsilon)$  constant factor approximation (for  $c_0 = 1$ ).



## Towards such a result (cont'd)

---

We would like to get a sampling and rescaling matrix  $S$  such that, simultaneously,

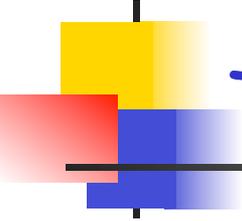
$$\sigma_{\min}(V_k^T S) \geq 1 - \sqrt{\frac{k}{c}} \quad \text{and} \quad \|(A - A_k) S\|_F \leq c_0 \|A - A_k\|_F$$

(for some small, fixed constant  $c_0$ ).

**Lamppost:** the work of Batson, Spielman, & Srivastava STOC 2009 (graph sparsification)

[We had to generalize their work to use a new barrier function which controls the [Frobenius and spectral norm of two matrices simultaneously](#). We then used a second phase to reduce the  $(2+\epsilon)$  approximation to  $(1+\epsilon)$ .]

We will omit these details, and instead [state the Batson, Spielman, & Srivastava STOC 2009 result as approximate matrix multiplication](#).



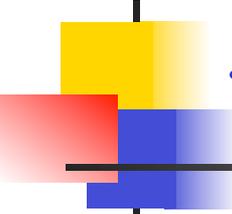
# The Batson-Spielman-Srivastava result

---

Let  $V_k$  be an  $n$ -by- $k$  matrix such that  $V_k^T V_k = I_k$ , with  $k < n$ , and let  $c$  be a sampling parameter (with  $c > k$ ).

There exists a deterministic algorithm which runs in  $O(cnk^3)$  time and constructs an  $n$ -by- $c$  sampling and rescaling matrix  $S$  such that

$$\|V_k^T V_k - V_k^T S S^T V_k\|_2 = \|I_k - V_k^T S S^T V_k\|_2 \leq \frac{k}{c}$$



# The Batson-Spielman-Srivastava result

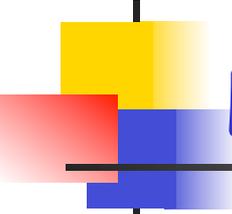
---

Let  $V_k$  be an  $n$ -by- $k$  matrix such that  $V_k^T V_k = I_k$ , with  $k < n$ , and let  $c$  be a sampling parameter (with  $c > k$ ).

There exists a deterministic algorithm which runs in  $O(cnk^3)$  time and constructs an  $n$ -by- $c$  sampling and rescaling matrix  $S$  such that

$$\|V_k^T V_k - V_k^T S S^T V_k\|_2 = \|I_k - V_k^T S S^T V_k\|_2 \leq \frac{k}{c}$$

- It is essentially a **matrix multiplication result!**
- Expensive to compute, but **very accurate and deterministic.**
- Works for **small values** of the sampling parameter  $c$ .
- The rescaling in  $S$  is **critical and non-trivial.**
- The algorithm is basically an **iterative, greedy approach** that uses two barrier functions to guarantee that the singular values of  $V_k^T S$  stay within boundaries.



# Lower bounds and alternative approaches

---

## Deshpande & Vempala, RANDOM 2006

A relative-error approximation necessitates at **least  $k/\epsilon$  columns**.

## Guruswami & Sinop, SODA 2012

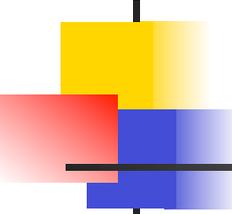
Alternative approaches, based on volume sampling, guarantee  
 **$(r+1)/(r+1-k)$  relative error bounds.**

This bound is asymptotically optimal (up to lower order terms).

The proposed **deterministic algorithm runs in  $O(rnm^3 \log m)$  time**, while the **randomized algorithm runs in  $O(rnm^2)$  time** and achieves the bound in expectation.

## Guruswami & Sinop, FOCS 2011

Applications of column-based reconstruction in Quadratic Integer Programming.

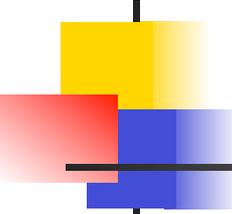


# Selecting rows/columns

---

## Selecting columns/rows from a matrix

- **Additive error** low-rank matrix approximation  
Frieze, Kannan, Vempala FOCS 1998, JACM 2004  
Drineas, Frieze, Kannan, Vempala, Vinay SODA 1999, JMLR 2004.
- **Relative-error** low-rank matrix approximation and least-squares problems  
Via **leverage scores** (Drineas, Mahoney, Muthukrishnan SODA 2006, SIMAX 2008)  
Via **volume sampling** (Deshpande, Rademacher, Vempala, Wang SODA 2006)
- **Efficient algorithms** with relative-error guarantees (theory)  
Random Projections and the Fast Johnson-Lindenstrauss Transform  
Sarlos FOCS 2006, Drineas, Mahoney, Muthukrishnan, & Sarlos NumMath 2011
- **Efficient algorithms** with relative-error guarantees (numerical implementations)  
Solving over- and under-constrained least-squares problems 4x faster than current state-of-the-art.  
Amazon EC2-type implementations with M. W. Mahoney, X. Meng, K. Clarkson, D. Woodruff, et al.  
Tygert & Rokhlin PNAS 2007, Avron, Maymounkov, Toledo SISC 2010, Meng, Saunders, Mahoney ArXiv 2011
- **Optimal** relative-error guarantees with matching lower bounds  
Relative-error accuracy with asymptotically optimal guarantees on the number of sampled columns.  
(Boutsidis, Drineas, Magdon-Ismail FOCS 2011, Guruswami and Sinop SODA 2012)



# Not covered in this tutorial: Element-wise sampling

---

**Element-wise sampling:** Can entry-wise sampling be as accurate as column-sampling?

**Prior work:** additive-error guarantees.

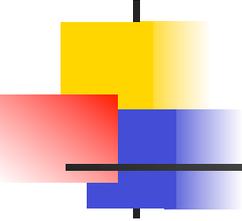
- To approximate a matrix  $A$ , keep a few elements of the matrix (instead of rows or columns) and zero out the remaining elements:

$$\tilde{A}_{ij} = \begin{cases} A_{ij}/p_{ij} & , \text{with probability } p_{ij} = \frac{A_{ij}^2}{\sum_{i,j} A_{ij}^2} \\ 0 & , \text{otherwise} \end{cases}$$

Compute a low-rank approximation to the sparse matrix  $\tilde{A}$  using iterative methods.

(Achlioptas & McSherry STOC 2001, JACM 2007; Drineas & Zouzias IPL 2011; Nguyen & Drineas ArXiv 2011)

- **Exact reconstruction possible** using uniform sampling for matrices that satisfy certain (strong) assumptions:  $A$  must be rank exactly  $k$ ,  $A$  must have uniform leverage scores (low coherence).  
(Candes & Recht 2008, Candes & Tao 2009, Recht 2009, Negahban & Wainwright 2010)
- Exact reconstruction needs **Trace Minimization** (the convex relaxation of Rank Minimization); some generalizations in the presence of well-behaved noise exist.

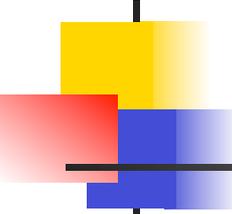


# Element-wise sampling (cont'd)

---

## Goals:

- Identify an **entry-wise probability distribution** (element-wise leverage scores) that achieves **relative-error accuracies** for approximating singular values and singular vectors, reconstructing the matrix, identifying influential entries in a matrix, solving least-squares problems, etc.
- Compute this probability distribution efficiently.
- Reconstruct the matrix in  $O(mn \text{ polylog}(m,n))$  time instead of using trace minimization methods.
- Prove matching lower bounds and provide high quality numerical implementations.



# Not covered in this tutorial: Solving systems of linear equations

---

**Solving systems of linear equations:** given an  $m$ -by- $n$  matrix  $A$  and an  $m$ -vector  $b$ , compute:

$$x_{opt} = \arg \min_{x \in \mathbb{R}^n} \|Ax - b\|_2$$

**Prior work:** relative-error guarantees for  $m \gg n$  or  $m \ll n$  and dense matrices  $A$ .

**Running time:**  $O(mn \text{ polylog}(n))$

(Drineas, Mahoney, Muthukrishnan, Sarlos NumMath 2011; improved by Boutsidis & Gittens ArXiv 2012)

**Main tool:** random projections via the Fast Hadamard Transform

**Numerical implementations:** Blendenpik (Avron, Maymounkov, Toledo SISC 2010)

**Still open:** sparse input matrices, some recent progress

(Mahoney & Meng, Clarkson & Woodruff STOC 2013)

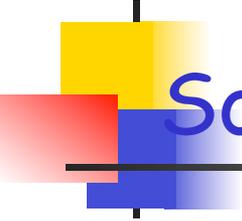
**Prior work:** relative-error guarantees for  $m = n$  and  $A$  Laplacian or SDD (symmetric diagonally dominant).

**Running time:**  $O(\text{nnz}(A) \log(n))$ , almost optimal bounds.

(Spielman, Teng & collaborators, many papers over the past 8 years, Koutis, Miller, Peng FOCS 2010 & FOCS 2011)

**Main tool:** Sparsify the input graph (Laplacian matrix) by sampling a subset of its edges with respect to a probability distribution called "effective resistances." Form (recursively) a preconditioning chain use Chebyshev's preconditioner.

**Graph Sparsification Step:** Koutis et al. approximated effective resistances via low-stretch spanning trees; Drineas & Mahoney Arxiv 2010 connected effective resistances to leverage scores.



# Solving systems of linear equations (cont'd)

---

**Fact:** in prior work graphs are fundamental: effective resistances, low-stretch spanning trees, etc.

## Goals:

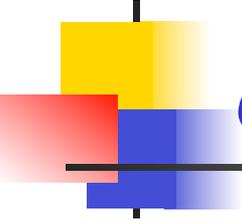
Move current state-of-the-art beyond Laplacians and SDD matrices to - say - SPD (positive semidefinite) matrices.

Paradigm shift: *deterministic accuracy guarantees with a randomized running time*, as opposed to probabilistic accuracy with deterministic running times.

If  $\epsilon$  is the target relative error, the running time should depend on  $\text{poly}(\log(1/\epsilon))$  instead of  $\text{poly}(1/\epsilon)$ .

In Theoretical Computer Science, most algorithms achieve the latter guarantee.

In Numerical Linear Algebra the former dependency is always the objective.



# Conclusions

---

- Randomization and sampling can be used to solve problems that are massive and/or computationally expensive.
- By (carefully) sampling rows/columns of a matrix, we can construct new sparse/smaller matrices that behave like the original matrix.
- By preprocessing the matrix using random projections, we can sample rows/columns (of the preprocessed matrix) uniformly at random and still get nice “behavior”.