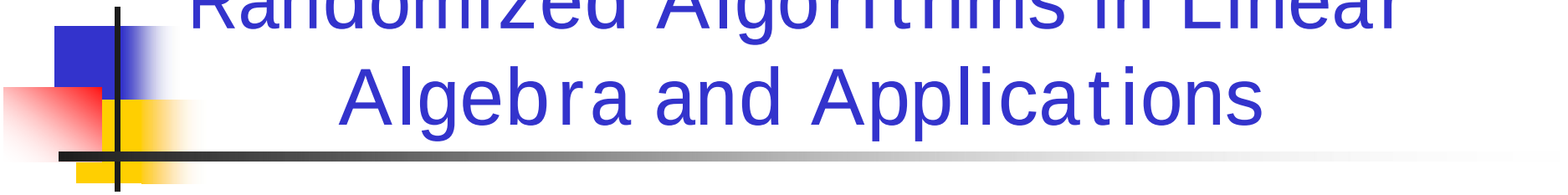


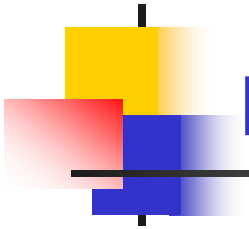
Randomized Algorithms in Linear Algebra and Applications

Petros Drineas

Rensselaer Polytechnic Institute
Computer Science Department

To access my web page:

Google drineas



Randomized algorithms

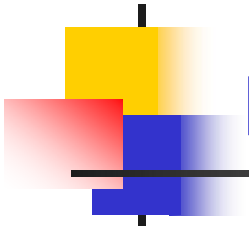
Randomization and sampling allow us to design provably accurate algorithms for problems that are:

- Massive

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

- Computationally expensive or even NP-hard

(e.g., combinatorial problems such as the Column Subset Selection Problem)



Mathematical background

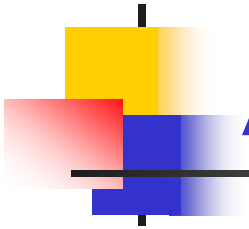
Obviously, **linear algebra** and **probability theory**. More specifically,

- **Ideas underlying the design and analysis of randomized algorithms**

E.g., the material covered in chapters 3, 4, and 5 of the "Randomized Algorithms" book of Motwani and Raghavan.

- **Matrix perturbation theory**

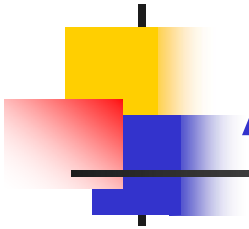
E.g., take a look at "Matrix Perturbation Theory" by Stewart and Sun, or "Matrix Analysis" by R. Bhatia.



Applying the math background

- **Randomized algorithms**
  - By (carefully) sampling rows/columns/entries of a matrix, we can construct new matrices (that have smaller dimensions or are sparse) and have bounded distance (in terms of some matrix norm) from the original matrix (with some failure probability).
  - By preprocessing the matrix using random projections (*), we can sample rows/columns/entries(?) much less carefully (uniformly at random) and still get nice bounds (with some failure probability).

(*) Alternatively, we can assume that the matrix is “well-behaved” and thus uniform sampling will work.



Applying the math background

- **Randomized algorithms**
  - By (carefully) sampling rows/columns/entries of a matrix, we can construct new matrices (that have smaller dimensions or are sparse) and have bounded distance (in terms of some matrix norm) from the original matrix (with some failure probability).
  - By preprocessing the matrix using random projections, we can sample rows/columns/entries(?) much less carefully (uniformly at random) and still get nice bounds (with some failure probability).
- **Matrix perturbation theory**
  - The resulting smaller/sparser matrices behave similarly (in terms of singular values and singular vectors) to the original matrices thanks to the norm bounds.

In this talk, I will illustrate a few "Randomized Algorithms" ideas that have been leveraged in the analysis of randomized algorithms in linear algebra.

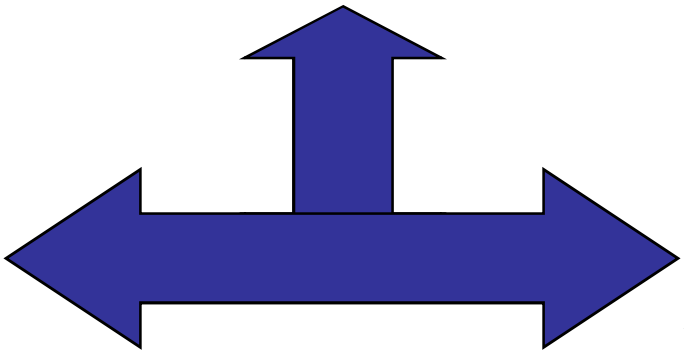
Interplay

(Data Mining) Applications

Biology & Medicine: **population genetics (coming up…)**

Electrical Engineering: testing of electronic circuits

Internet Data: recommendation systems, document-term data



Theoretical Computer Science

Randomized and approximation algorithms

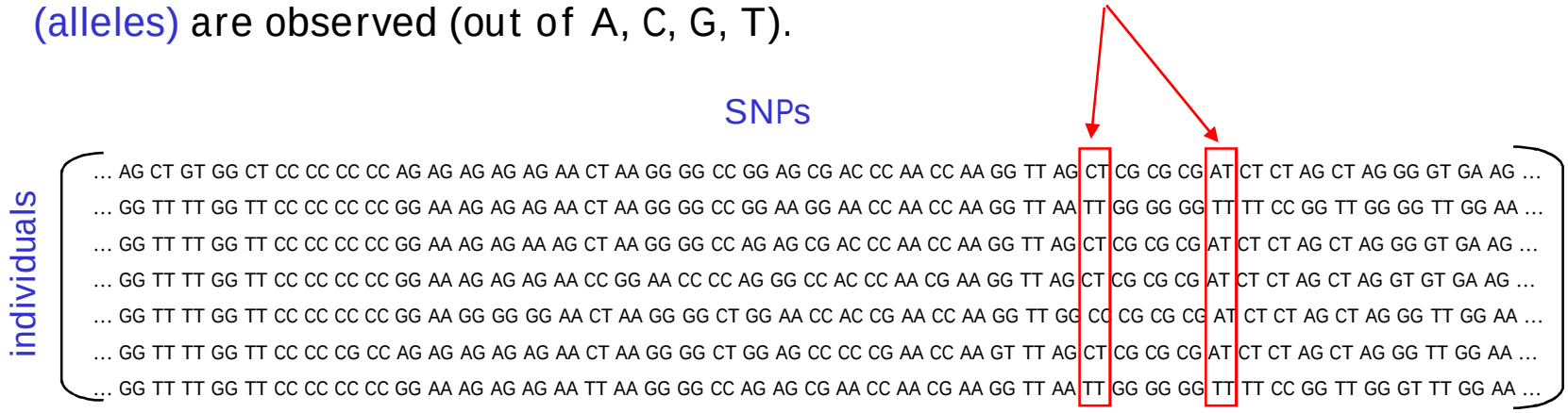
Numerical Linear Algebra

Matrix computations and Linear Algebra (ie., perturbation theory)

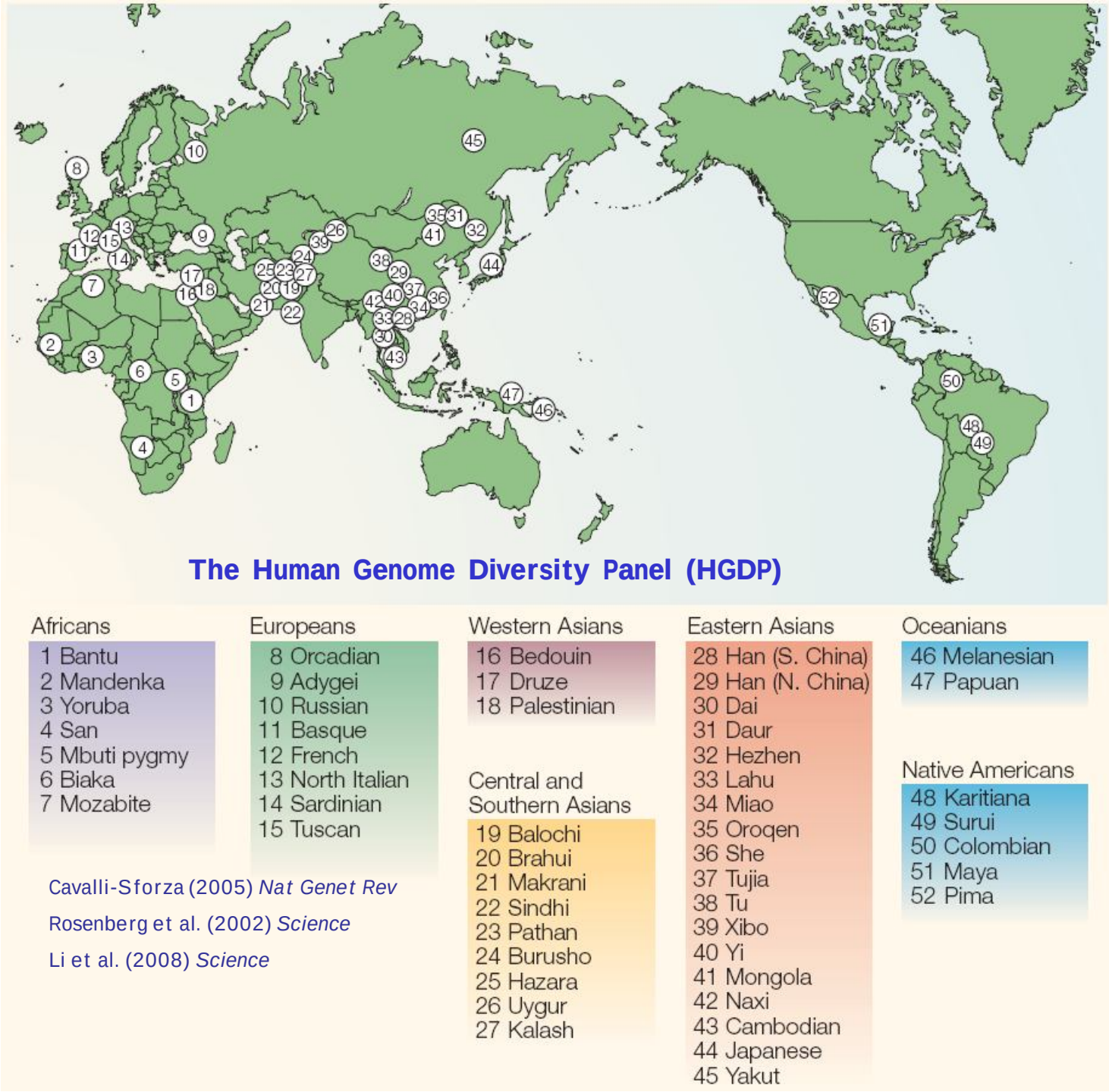
Human genetics

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).



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



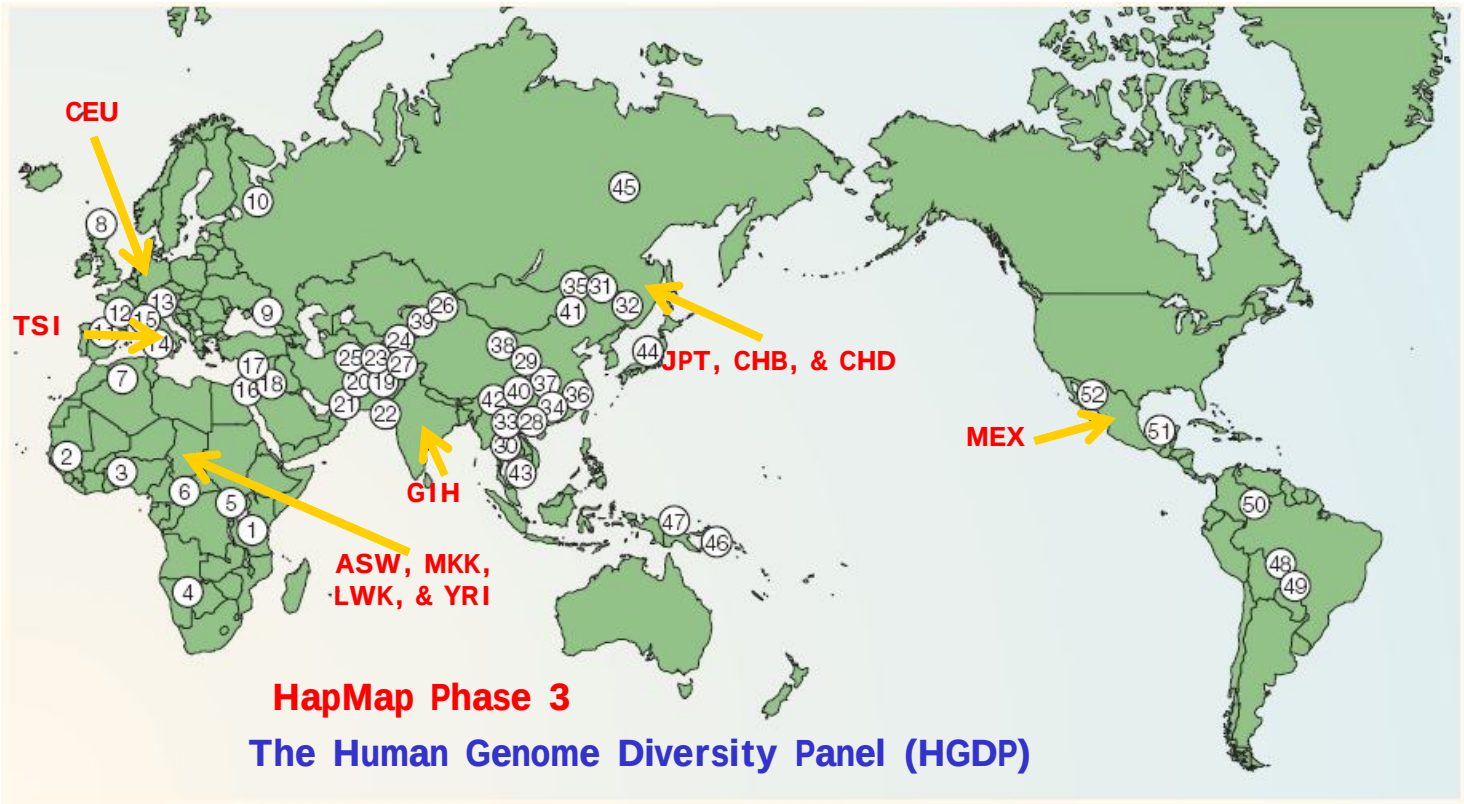
Cavalli-Sforza (2005) *Nat Genet Rev*

Rosenberg et al. (2002) *Science*

Li et al. (2008) *Science*

HGDP data

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

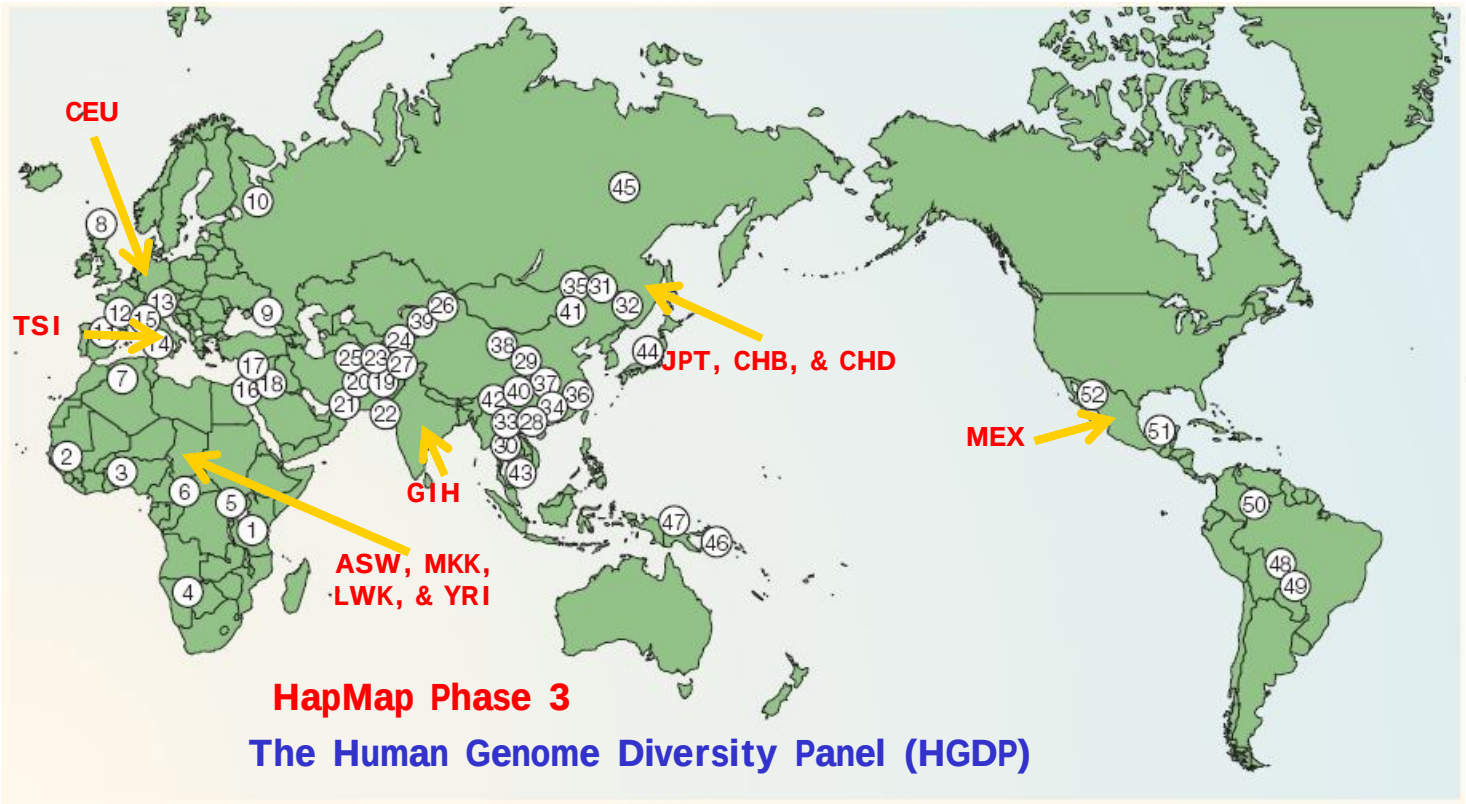
The International HapMap Consortium (2003, 2005, 2007) *Nature*

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*

HGDP data

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

HapMap Phase 3 data

- 1,207 samples
- 11 populations

We will apply SVD/PCA on the (joint) HGDP and HapMap Phase 3 data.

Matrix dimensions:

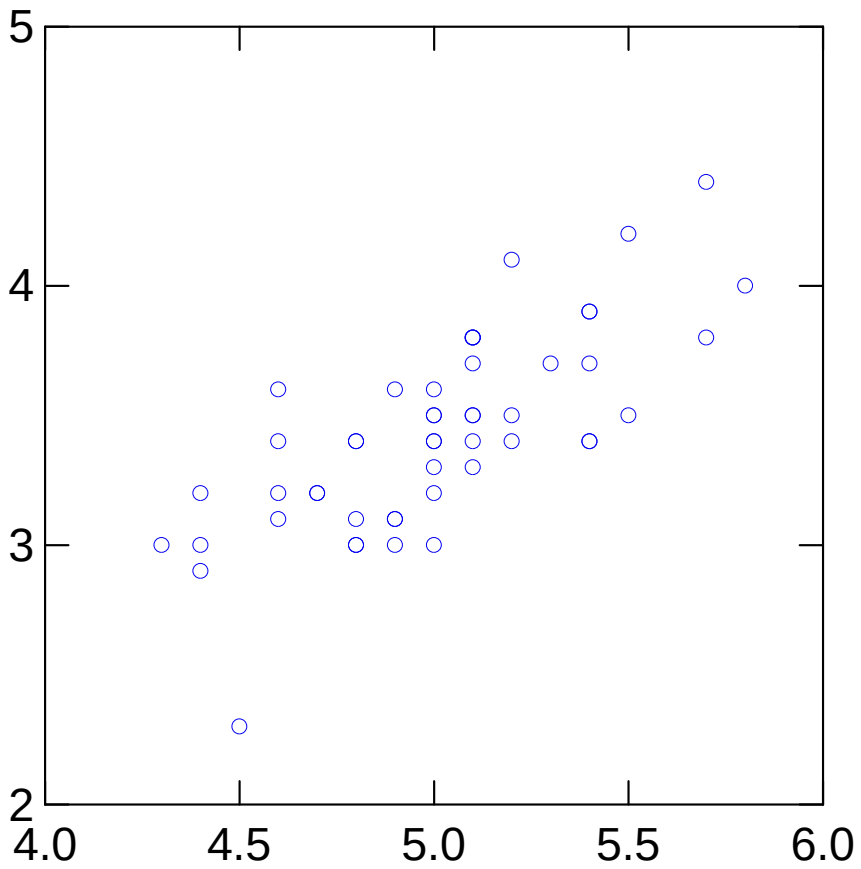
2,240 subjects (rows)

447,143 SNPs (columns)

Dense matrix:

over one billion entries

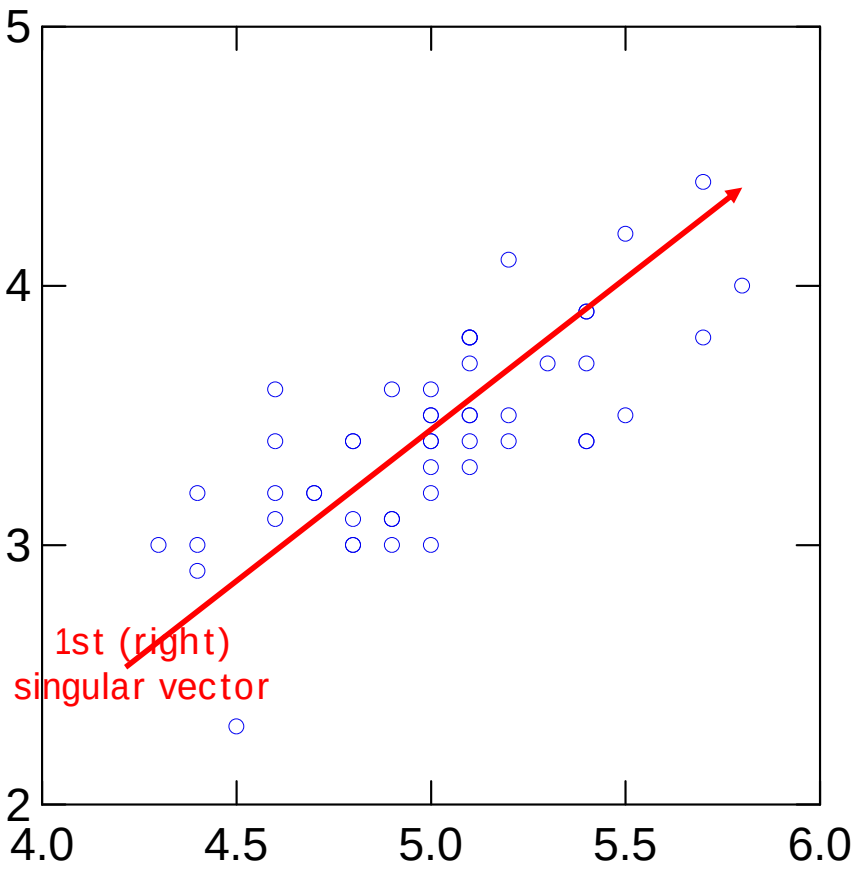
The Singular Value Decomposition (SVD)



Let the blue circles represent *m* data points in a 2-D Euclidean space.

Then, the SVD of the *m-by-2* matrix of the data will return …

The Singular Value Decomposition (SVD)



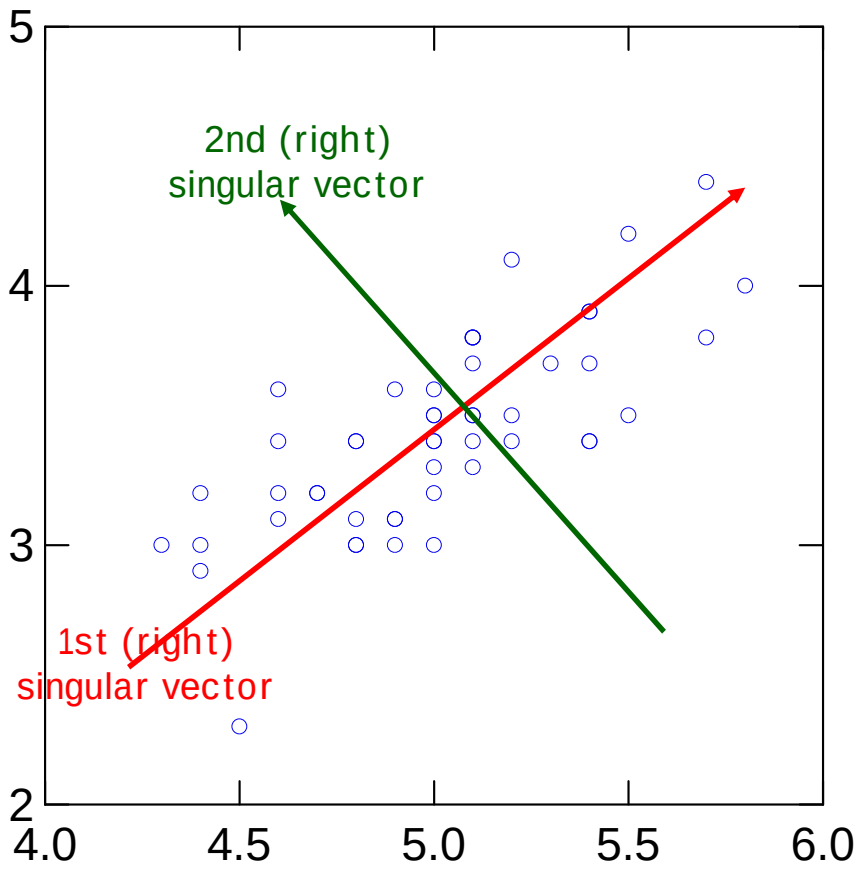
Let the blue circles represent *m* data points in a 2-D Euclidean space.

Then, the SVD of the *m-by-2* matrix of the data will return …

1st (right) singular vector:

direction of maximal variance,

The Singular Value Decomposition (SVD)



Let the blue circles represent *m* data points in a 2-D Euclidean space.

Then, the SVD of the *m-by-2* matrix of the data will return …

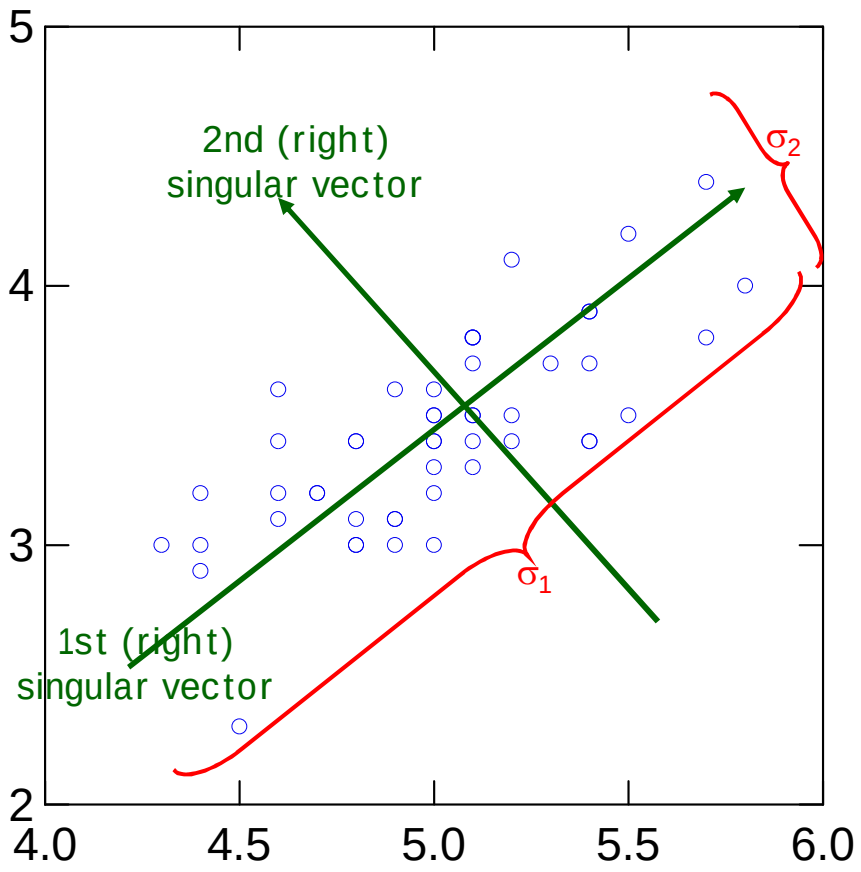
1st (right) singular vector:

direction of maximal variance,

2nd (right) singular vector:

direction of maximal variance, after removing the projection of the data along the first singular vector.

Singular values

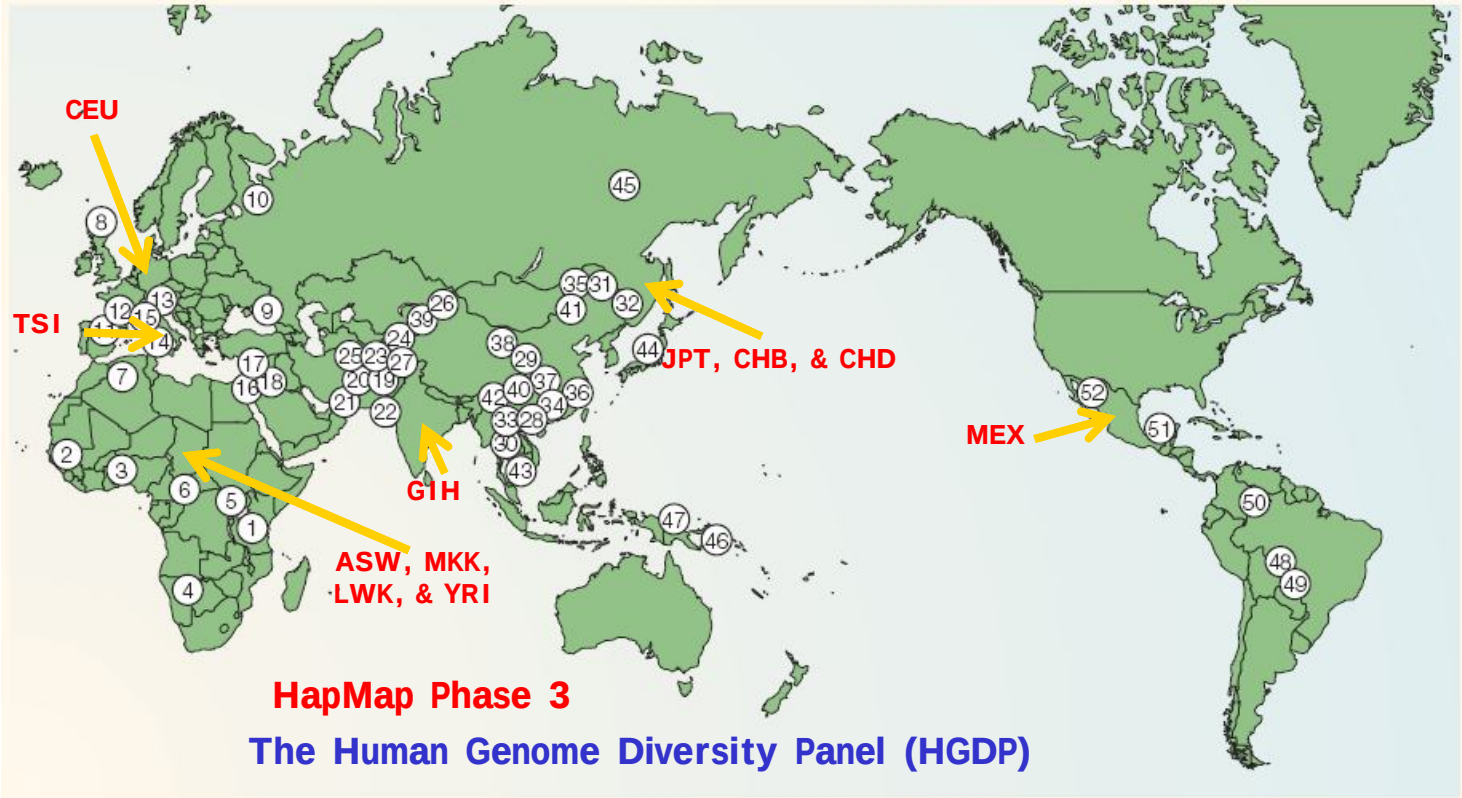


σ_1: measures how much of the data variance is explained by the first singular vector.

σ_2: measures how much of the data variance is explained by the second singular vector.

Principal Components Analysis (PCA) is done via the computation of the Singular Value Decomposition (SVD) of a (mean-centered) covariance matrix.

Typically, a small constant number (say k) of the top singular vectors and values are kept.



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*

HGDP data

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

HapMap Phase 3 data

- 1,207 samples
- 11 populations

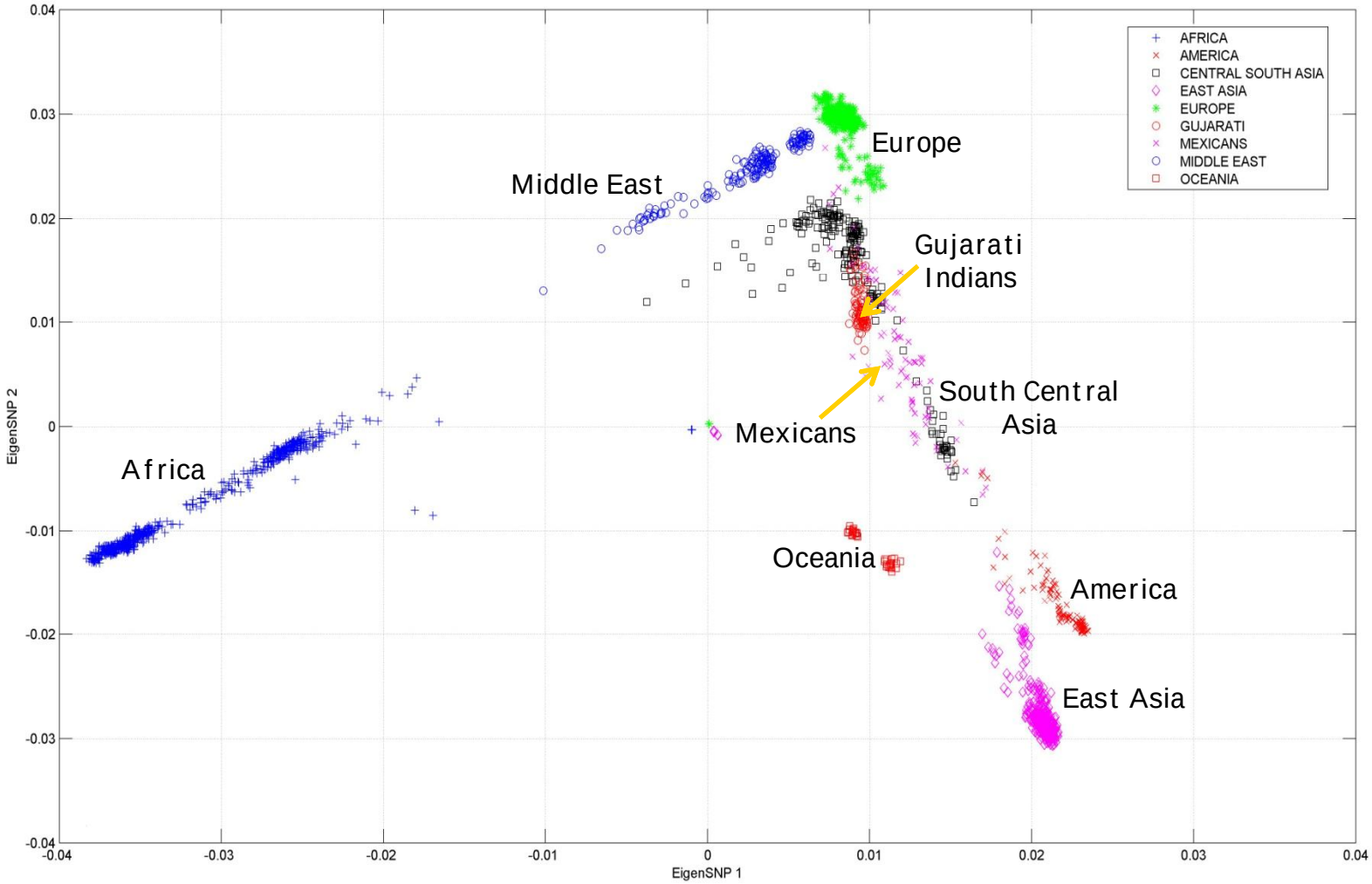
Matrix dimensions:

2,240 subjects (rows)

447,143 SNPs (columns)

SVD/PCA returns...

Paschou, Lewis, Javed, & Drineas (2010) J Med Genet

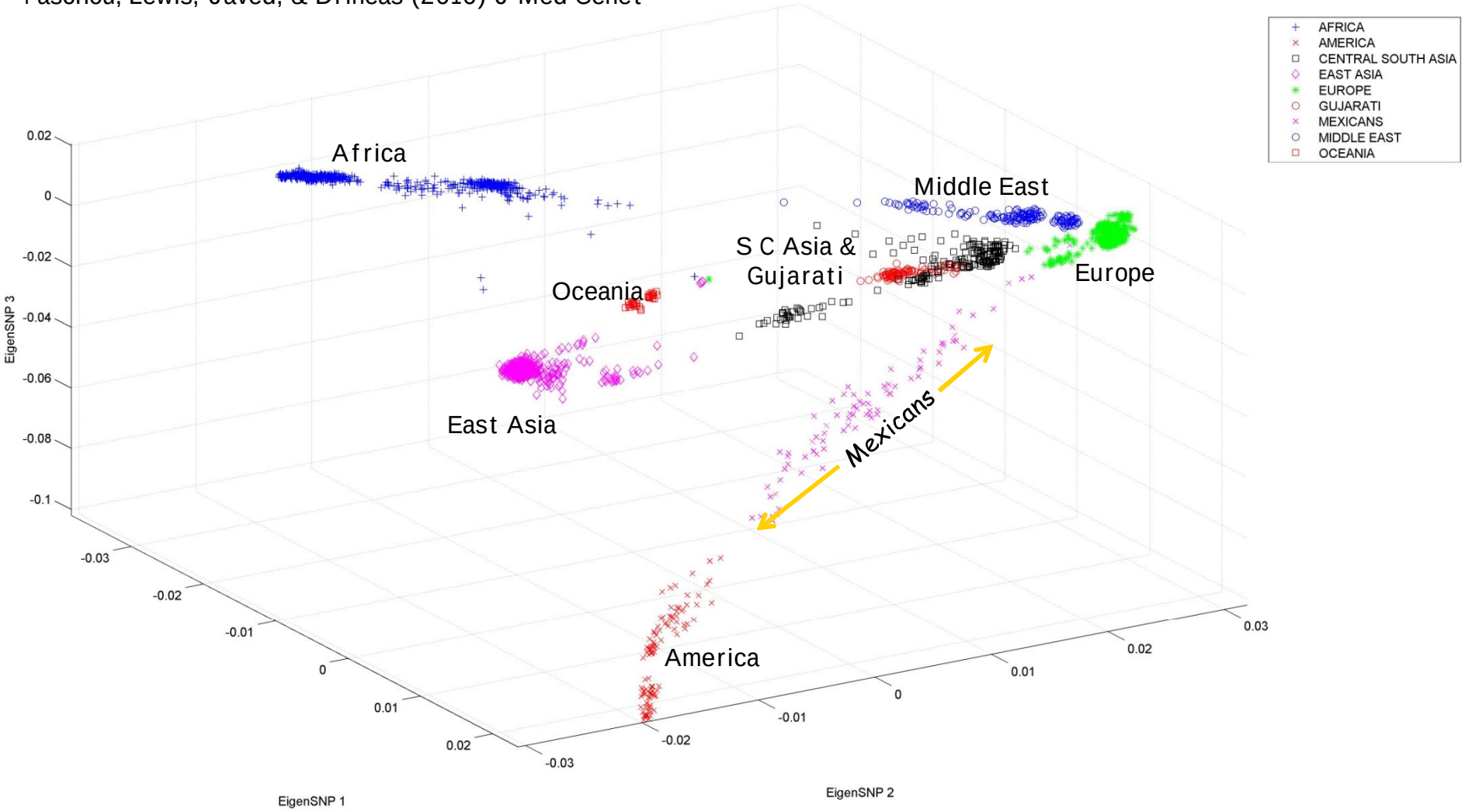


- Top two Principal Components (PCs or eigenSNPs)

(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.

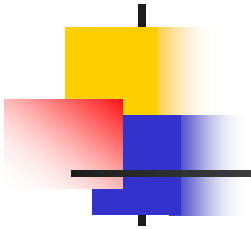
Paschou, Lewis, Javed, & Drineas (2010) J Med Genet



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?

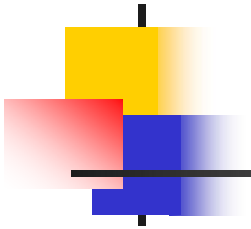
Formally: spanning the same subspace.



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 20 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 SVD of AA^T.
  - In a similar experiment, we computed **1,200 SVDs** on matrices of dimensions (approx.) 1,200-by-450,000 (roughly speaking a full leave-one-out cross-validation experiment).

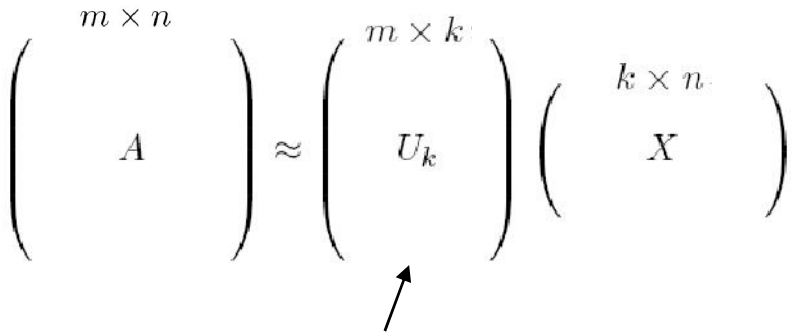
  (Drineas, Lewis, & Paschou (2010) PLoS ONE, in press)
- **Obviously, running time is a concern.**
- **We need efficient, easy to implement, methods.**



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.

SVD decomposes a matrix as...

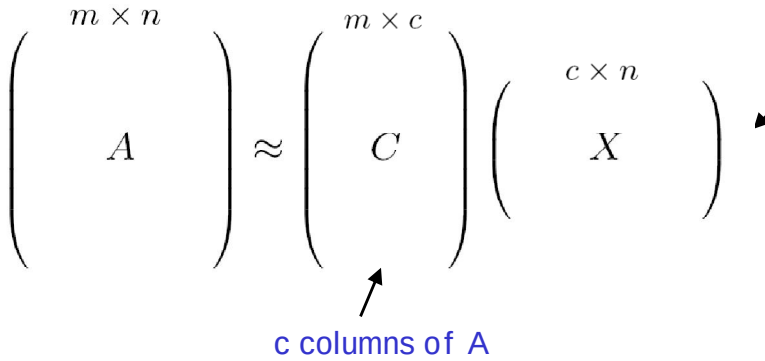


The SVD has strong optimality properties.

- 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

Drineas, Mahoney, & Muthukrishnan (2008) SIAM J Mat Anal Appl
Mahoney & Drineas (2009) PNAS



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

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

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

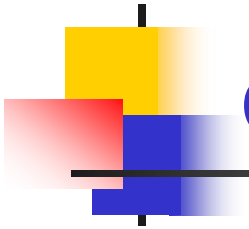
c columns of A

Easy to prove that optimal X = $C^{+}A$. (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 hard combinatorial problem, closely related to the so-called Column Subset Selection Problem (CSSP).

The CSSP has been heavily studied in Numerical Linear Algebra.



Our perspective

The two issues are connected

- There exist "good" columns in any matrix that contain information about the top principal components.
- We can identify such columns via a simple statistic: **the leverage scores.**
- This does not immediately imply faster algorithms for the SVD, but, combined with random projections, it does!

Key mathematical apparatus: approximating matrix multiplication.

Approximating Matrix Multiplication …

Frieze, Kannan, & Vempala (1998) FOCS, (2004) JACM
Drineas, Kannan, & Mahoney (2006) SICOMP

Problem Statement

Given an m-by-n matrix A and an n-by-p matrix B, approximate the product A·B,

OR, equivalently,

Approximate the sum of *n* rank-one matrices.

$$A \cdot B = \sum_{i=1}^{n} \underbrace{\left(A^{(i)} \right) \cdot \left(B_{(i)} \right)}_{\in \mathcal{R}^{m \times p}}$$

i-th column of A

i-th row of B

Each term in the summation is a rank-one matrix

...by sampling

$$A \cdot B = \sum_{i=1}^{n} \underbrace{\left(A^{(i)} \right) \cdot \left(B_{(i)} \right)}_{\in \mathcal{R}^{m \times p}}$$

i-th column of A

i-th row of B

Algorithm

1. Fix a set of probabilities p_i, i=1...n, summing up to 1.
2. For t=1 up to s,

   set j_t = i, where $Pr(j_t = i) = p_i$;

   (Pick s terms of the sum, with replacement, with respect to the p_i.)
3. Approximate the product AB by summing the s terms, after scaling.

Sampling (cont'd)

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

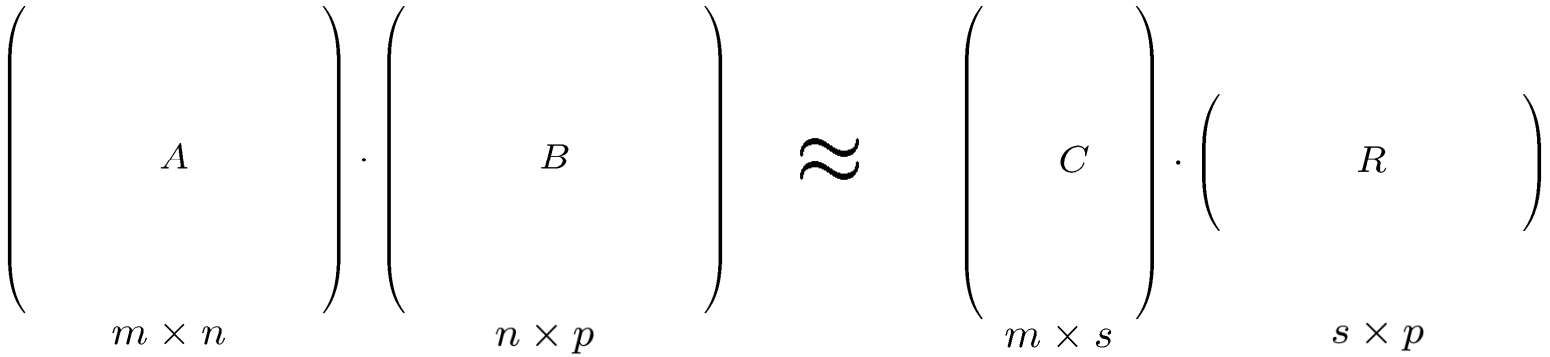
i-th column of A

i-th row of B

$$\approx \frac{1}{s} \sum_{t=1}^{s} \frac{1}{p_{j_t}} \underbrace{\begin{pmatrix} \\ A^{(j_t)} \\ \\ \end{pmatrix} \cdot \begin{pmatrix} \quad B_{(j_t)} \quad \end{pmatrix}}_{\in \mathcal{R}^{m \times p}}$$

Keeping the terms $j_1, j_2, \ldots j_s$.

The algorithm (matrix notation)



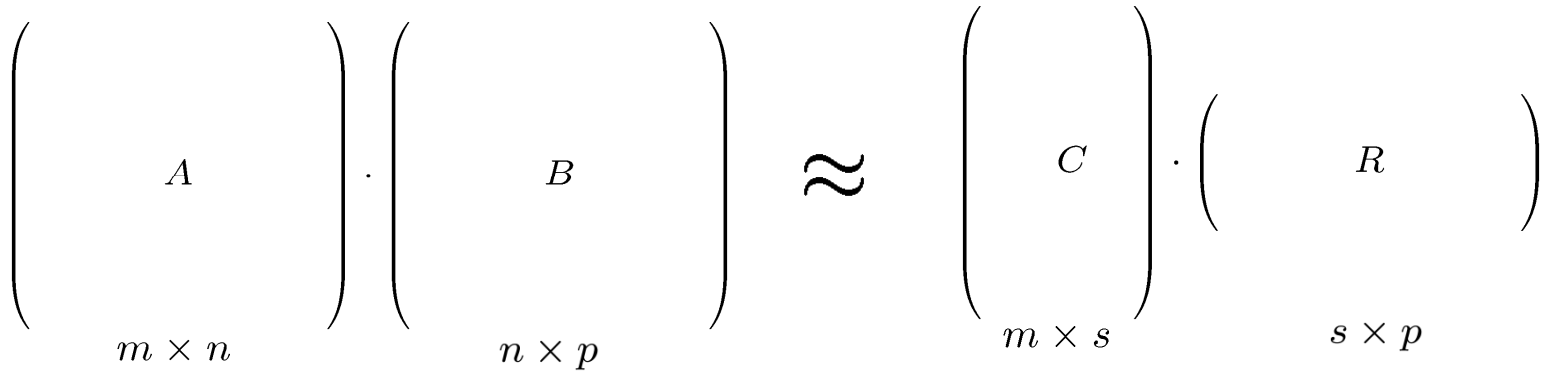
Algorithm

1. Pick s columns of A to form an m-by-s matrix C and the corresponding s rows of B to form an s-by-p matrix R.
2. (discard A and B) Approximate A · B by C · 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).

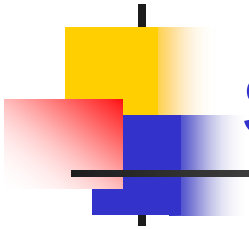
The algorithm (matrix notation, cont'd)



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

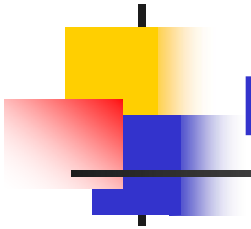
$$\Pr(j_t = i) = \frac{\left|A^{(i)}\right|\left|B_{(i)}\right|}{\sum_{i=1}^{n}\left|A^{(i)}\right|\left|B_{(i)}\right|}$$

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



Simple Lemmas ...

- The expectation of CR (element-wise) is AB.
- Our adaptive sampling minimizes the variance of the estimator.
- It is easy to implement the sampling in two passes.



Error Bounds

For the above algorithm,

$$E\left(\|AB - CR\|_{2,F}\right) \;\leq\; \frac{1}{\sqrt{s}}\|A\|_F\|B\|_F$$

$$\|X\|_F^2 = \sum_{i,j} X_{ij}^2 \quad , \quad \|X\|_2 = \max_{x:|x|=1} |Xx|$$

Tight concentration bounds can be proven via a martingale argument.

Special case: B = A^T

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

$$\Pr(\text{picking } i) = \frac{\left|A^{(i)}\right|^2}{\sum_{i=1}^{n}\left|A^{(i)}\right|^2} = \frac{\left|A^{(i)}\right|^2}{\|A\|_F^2}$$

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

$$E\left(\left\|AA^T - CC^T\right\|_{2,F}\right) \leq \frac{1}{\sqrt{s}}\|A\|_F^2$$

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

Rudelson and Virshynin (2007) JACM

A stronger result for the spectral norm is proven by M. Rudelson and R. Vershynin.

Assume (for normalization) that $||A||_2 = 1$. If

$$s = O\left(\frac{\|A\|_F^2}{\epsilon^2}\log\left(\frac{\|A\|_F^2}{\epsilon^2}\right)\right)$$

then for any $0 < \varepsilon < 1$

$$\mathbf{E}\left[\left\|AA^T - CC^T\right\|_2\right] \leq \epsilon$$

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

Rudelson and Virshynin (2007) JACM

A stronger result for the spectral norm is proven by M. Rudelson and R. Vershynin

Assume (for normalization) that $||A||_2 = 1$. If

$$s = O\left(\frac{\|A\|_F^2}{\epsilon^2}\log\left(\frac{\|A\|_F^2}{\epsilon^2}\right)\right)$$

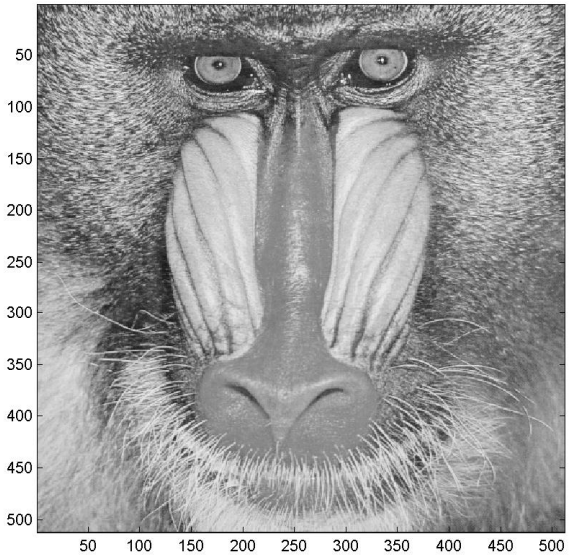
then for any $0 < \varepsilon < 1$

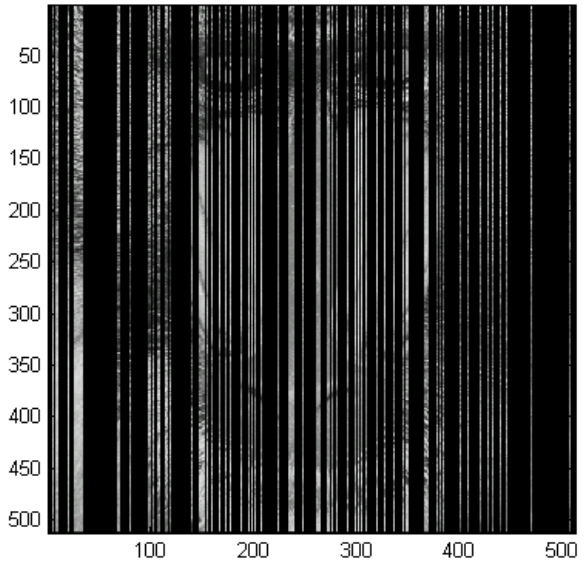
$$\mathbf{E}\left[\left\|AA^T - CC^T\right\|_2\right] \leq \epsilon$$

- Uses a beautiful result of M. Rudelson for random vectors in isotropic position and Talagrand's measure concentration results.
- Similar results can also be proven using recent results on matrix-valued Chernoff bounds.

See Ahlswede and Winter (2001), Recht (2009), Oliveira (2010).

Approximating singular vectors





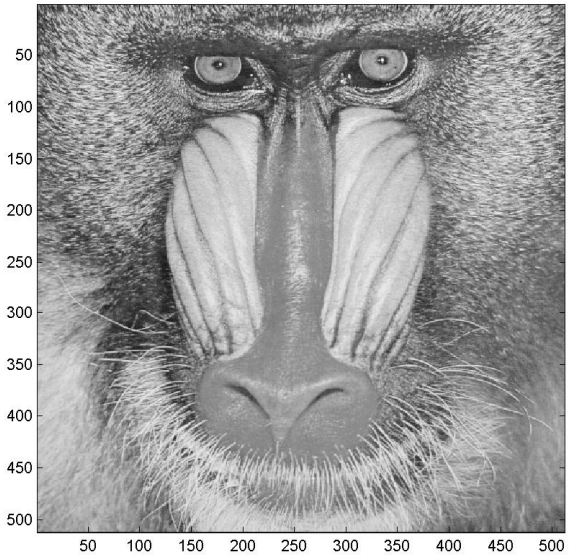
Original matrix

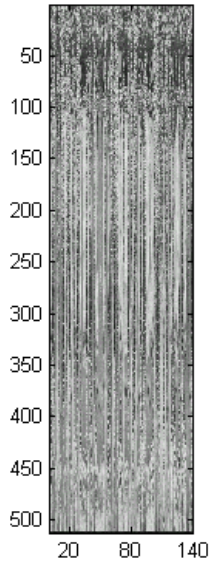
Sampling (s=140 columns)

1. Sample s (=140) columns of the original matrix A and rescale them appropriately to form a 512-by-c matrix C.
2. Project A on CC^+ and show that $A-CC^+A$ is “small”.

(C^+ is the pseudoinverse of C)

Approximating singular vectors





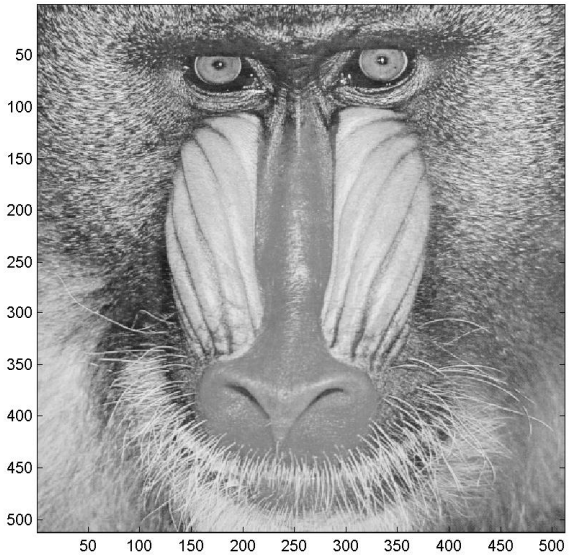
Original matrix

Sampling (s=140 columns)

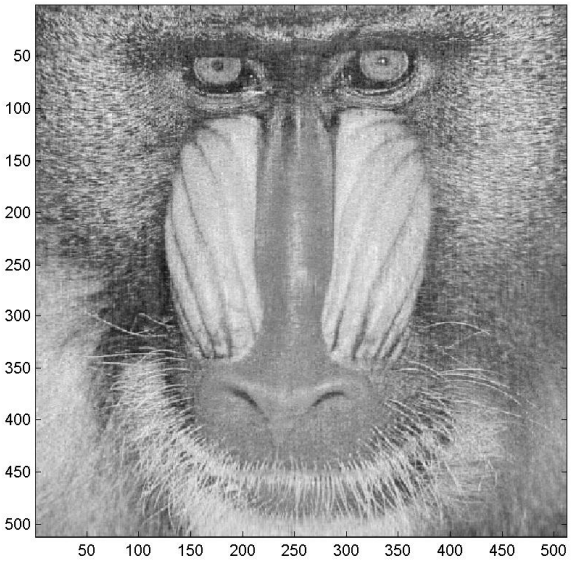
1. Sample s (=140) columns of the original matrix A and rescale them appropriately to form a 512-by-c matrix C.
2. Project A on CC^+ and show that $A-CC^+A$ is "small".

(C^+ is the pseudoinverse of C)

Approximating singular vectors (cont'd)



A



$CC^{+}A$

The fact that $AA^{T} - CC^{T}$ is small will imply that $A\text{-}CC^{+}A$ is small as well.

Proof (spectral norm)

Using the triangle inequality and properties of norms,

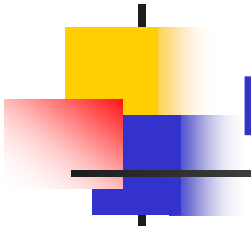
$$\begin{aligned}
\left\|A - CC^{\dagger}A\right\|_2^2 &= \left\|\left(I - CC^{\dagger}\right)A\right\|_2^2 \\
&= \left\|\left(I - CC^{\dagger}\right)AA^T\left(I - CC^{\dagger}\right)^T\right\|_2 \\
&\leq \left\|\left(I - CC^{\dagger}\right)\left(AA^T - CC^T\right)\left(I - CC^{\dagger}\right)^T\right\|_2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

Proof (spectral norm, cont'd)

Using the triangle inequality and properties of norms,

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

projector matrices



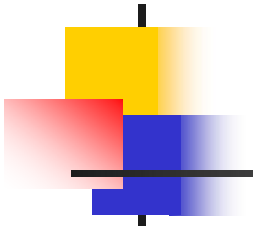
Proof (spectral norm), cont'd

We can now use our matrix multiplication result:

$$\begin{aligned}\mathbf{E}\left[\left\|A - CC^{\dagger}A\right\|_2\right] &\leq \mathbf{E}\left[\left\|AA^T - CC^T\right\|_2\right] \\ &\leq \frac{1}{\sqrt{s}}\|A\|_F\end{aligned}$$

(We could also have applied the Rudelson-Virshynin bound.)

Important: selecting the columns in this setting is trivial and can be implemented in a couple of (sequential) passes over the input matrix.



Is this a good bound?

$$\begin{aligned}\mathbf{E}\left[\left\|A - CC^{\dagger}A\right\|_2\right] &\leq \mathbf{E}\left[\left\|AA^T - CC^T\right\|_2\right] \\ &\leq \frac{1}{\sqrt{s}}\|A\|_F\end{aligned}$$

Problem 1: If s = n, we still 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 s=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 s=k and depend (roughly) on $(k(n-k))^{1/2}\ ||A-A_k||_2$

Relative-error Frobenius norm bounds

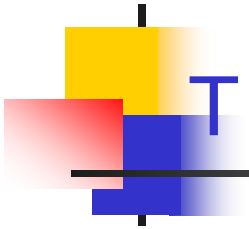
Drineas, Mahoney, & Muthukrishnan (2008) SIAM J Mat Anal Appl

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

at most $O((k/\varepsilon^2) \log (k/\varepsilon))$ columns of A

such that with probability at least .9

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



The algorithm

Input: m-by-n matrix A,

$0 < \varepsilon < .5$, the desired accuracy

Output: C, the matrix consisting of the selected columns

Sampling algorithm

- Compute probabilities p_j summing to 1
- Let $c = O((k/\varepsilon^2) \log (k/\varepsilon))$.
- 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

Subspace sampling (Frobenius norm)

$$\underset{m \times n}{\left(A_k \right)} = \underset{m \times k}{\left(U_k \right)} \cdot \underset{k \times k}{\left(\Sigma_k \right)} \cdot \underset{k \times n}{\left(V_k^T \right)}$$

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

S_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)

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

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

S_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

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

Normalization s.t. the p_j sum up to 1

Subspace sampling (Frobenius norm)

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

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

S_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.

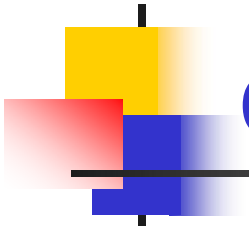
Subspace sampling in $O(mn^2)$ time

Leverage scores

(many references in the statistics community)

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

Normalization s.t. the p_j sum up to 1



Computing leverage scores efficiently

Problem

We do not know how to do it without computing the SVD: **computationally expensive.**

Open question: can we approximate **the leverage scores** efficiently?

Solution 1

Preprocess the matrix and make those scores uniform! Then sample uniformly at random.

We borrow ideas from the JL-transform literature: COMING UP.

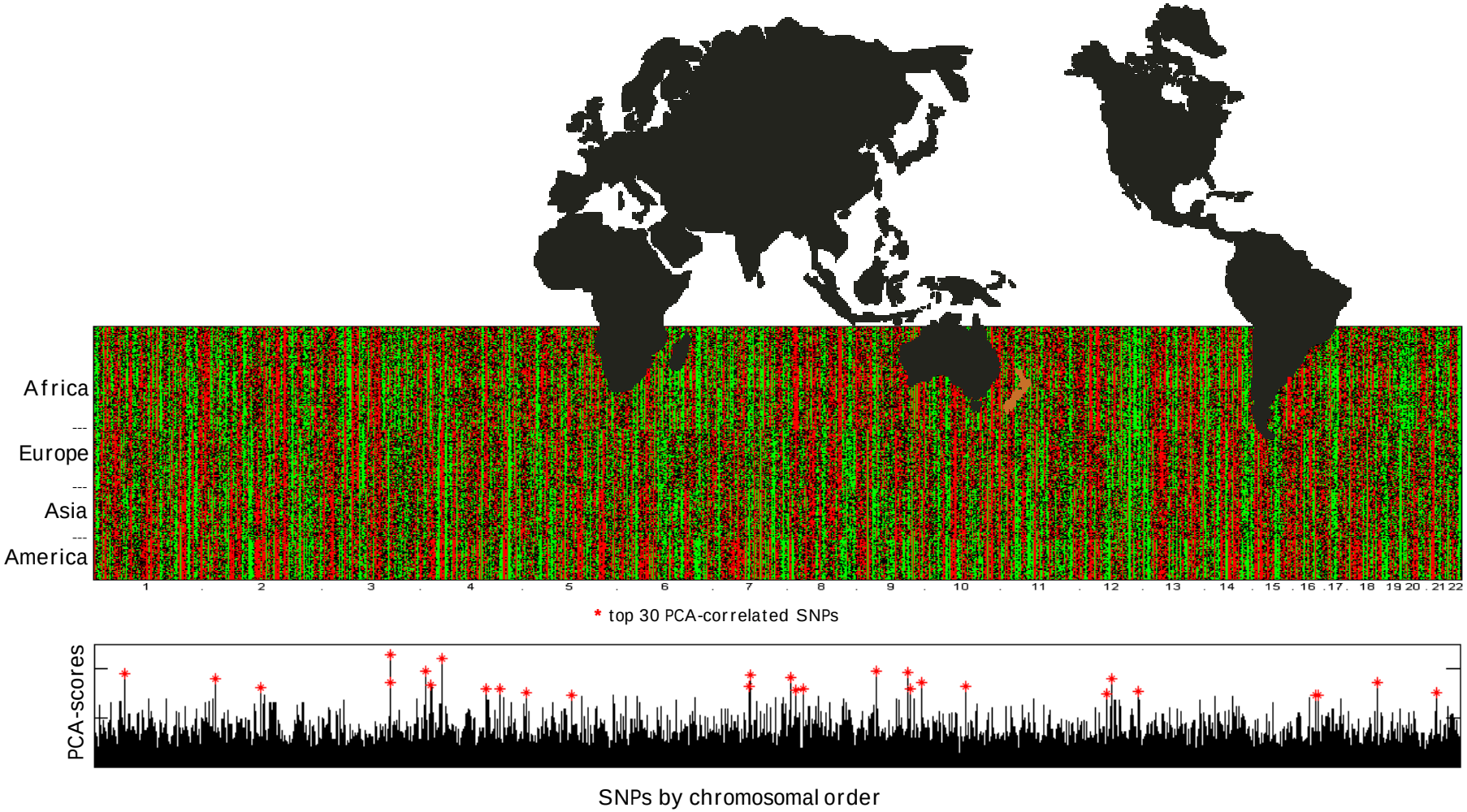
Solution 2

Use volume sampling-based methods to identify $O(k^2 \log k / \varepsilon^2)$ columns in $O(mnk^2)$ time.

See Deshpande and Vempala (2006) RANDOM.

BACK TO POPULATION GENETICS DATA

Selecting PCA SNPs for individual assignment to four continents (Africa, Europe, Asia, America)

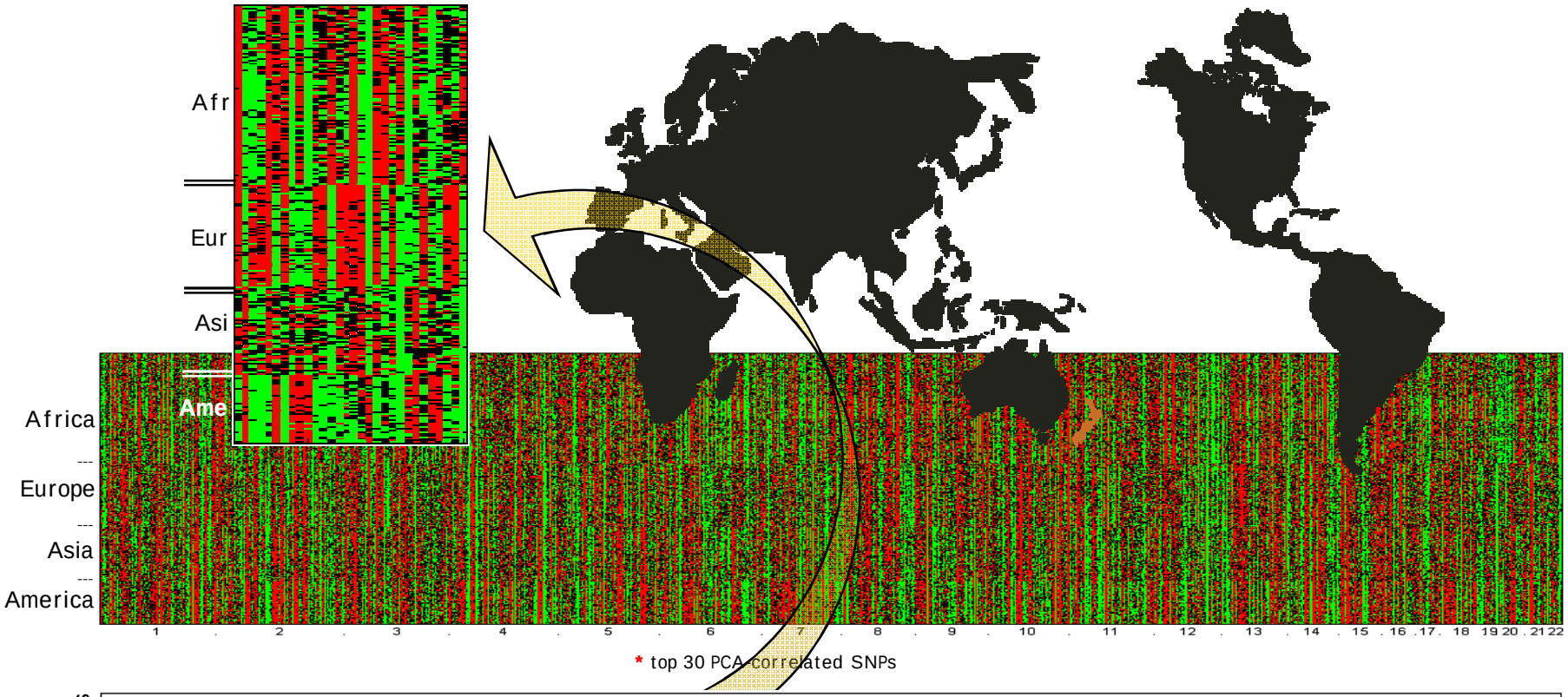


Paschou et al (2007; 2008) PLoS Genetics

Paschou et al (2010) J Med Genet, in press

Drineas et al (2010) PLoS One, in press

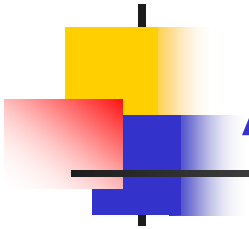
Selecting PCA SNPs for individual assignment to four continents (Africa, Europe, Asia, America)



Paschou et al (2007; 2008) PLoS Genetics

Paschou et al (2010) J Med Genet, in press

Drineas et al (2010) PLoS One, in press



An interesting observation

Sampling w.r.t. to leverage scores results in redundant columns being selected.

This is not surprising, since (almost) identical columns have (almost) the same leverage scores and thus might be all selected, even though they do not really add new "information."

First Solution:

Apply a "redundancy removal" step, by running some deterministic algorithm for the CSSP on the sampled columns.

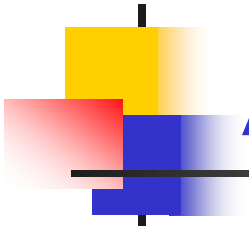
This works very well empirically, and even naïve CSSP algorithms (such as the pivoted QR factorization) return excellent results.

Conjecture:

The "leverage scores" filter out relevant columns, thus allowing deterministic methods to do a better job later.

See Paschou et al. (2008) PLoS Genetics for applications to population genetics.

See Boutsidis et al. (2009, 2010) SODA, ArXiv for theoretical foundations.



An interesting observation

Second Solution:

We can apply clustering to the sampled columns and then return a representative column from each cluster.

Very useful for SNP data since it permits clustering of SNPs that have similar functionalities and thus allows better understanding of the proposed ancestry-informative panels.

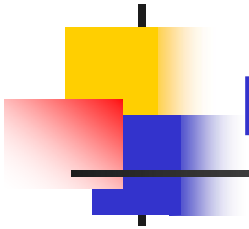
See Paschou et al (2010) J Med Genet.

Third Solution:

The (theoretically optimal) volume-sampling approach with marginal probabilities.

Some of its variants are based on random projections and seem useful for massive data.

See Deshpande and Rademacher (2010) ArXiv.



Random projections: the JL lemma

For every set S of m points in $\mathbb{R}^n$ and every $\epsilon > 0$, there exists a mapping $f : \mathbb{R}^n \to \mathbb{R}^s$, where $s = O\left(\log m/\epsilon^2\right)$, such that for all points $u \in S$,

$$(1-\epsilon)\ \|u\|_2 \leq\ \|f(u)\|_2 \leq (1+\epsilon)\ \|u\|_2$$

holds with probability at least $1 - 1/m^2$.

Johnson & Lindenstrauss (1984)

Random projections: the JL lemma

For every set S of m points in $\mathbb{R}^n$ and every $\epsilon > 0$, there exists a mapping $f : \mathbb{R}^n \to \mathbb{R}^s$, where $s = O\left(\log m/\epsilon^2\right)$, such that for all points $u \in S$,

$$(1-\epsilon)\ \|u\|_2 \leq\ \|f(u)\|_2 \leq (1+\epsilon)\ \|u\|_2$$

holds with probability at least $1 - 1/m^2$.

Johnson & Lindenstrauss (1984)

- We can represent S by an m-by-n matrix A, whose rows correspond to points.
- We can represent all f(u) by an m-by-s Ã.
- The "mapping" corresponds to the construction of an n-by-s matrix R and computing

$$\tilde{A} = AR$$

(The original JL lemma was proven by projecting the points of S to a random k-dimensional subspace.)

Different constructions for R

- Frankl & Maehara (1988): random orthogonal matrix
- DasGupta & Gupta (1999): matrix with entries from N(0,1), normalized
- Indyk & Motwani (1998): matrix with entries from N(0,1)
- **Achlioptas (2003):** matrix with entries in {-1,0,+1}
- Alon (2003): optimal dependency on n, and almost optimal dependency on ε

Construct an n-by-s matrix R such that:

$$R_{ij} = \sqrt{3} \times \begin{cases} +1 & ,\text{w.p. } 1/6 \\ 0 & ,\text{w.p. } 2/3 \\ -1 & ,\text{w.p. } 1/6 \end{cases}$$

Return: $\tilde{A} = \frac{1}{\sqrt{s}} AR \in \mathbb{R}^{m \times s}$

O(mns) = O(mn logm / ε^2) time computation

Fast JL transform

Ailon & Chazelle (2006) FOCS, Matousek (2006)

$$P \in \mathbb{R}^{s \times n}$$

$$s = O\left(\log m/\epsilon^2\right)$$

$$P_{ij} = \sqrt{q} \times \begin{cases} +1 & \text{,w.p. } q/2 \\ 0 & \text{,w.p. } 1\text{-}q \\ -1 & \text{,w.p. } q/2 \end{cases}$$

$$q = O\left(\frac{\log^2 m}{n}\right)$$

$$H \in \mathbb{R}^{n \times n}$$

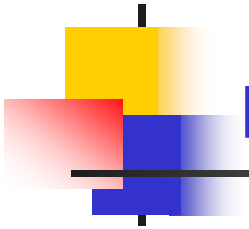
Normalized Hadamard-Walsh transform matrix

(if n is not a power of 2, add all-zero columns to A)

$$D \in \mathbb{R}^{n \times n}$$

Diagonal matrix with D_{ii} set to +1 or -1 w.p. ½.

$$R = (PHD)^T \in \mathbb{R}^{n \times s} \longrightarrow \tilde{A} = \frac{1}{\sqrt{s}} AR$$



Fast JL transform, cont'd

Applying PHD on a vector u *in* R^n is fast, since:

- Du : $O(n)$, since D is diagonal,
- H(Du) : $O(n \log n)$, using the Hadamard-Walsh algorithm,
- P(H(Du)) : $O(\log^3 m/\varepsilon^2)$, since P has on average $O(\log^2 n)$ non-zeros per row (in expectation).

Back to approximating singular vectors

Let A by an m-by-n matrix whose SVD is:

$$A = U\Sigma V^T \in \mathbb{R}^{m\times n}$$

Apply the (HD) part of the (PHD) transform to A.

$$ADH = U\Sigma \left(V^T DH\right) \in \mathbb{R}^{m\times n}$$

orthogonal matrix

Observations:

1. The left singular vectors of ADH span the same space as the left singular vectors of A.
2. The matrix ADH has (up to log n factors) uniform leverage scores .

   (Thanks to V^THD having bounded entries – the proof closely follows JL-type proofs.)
3. We can approximate the left singular vectors of ADH (and thus the left singular vectors of A) by uniformly sampling columns of ADH.

Back to approximating singular vectors

Let A by an m-by-n matrix whose SVD is:

$$A = U\Sigma V^T \in \mathbb{R}^{m\times n}$$

Apply the (HD) part of the (PHD) transform to A.

$$ADH = U\Sigma \left(V^T DH\right) \in \mathbb{R}^{m\times n}$$

orthogonal matrix

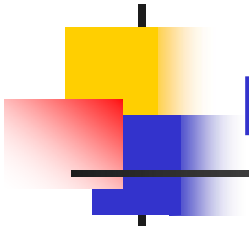
Observations:

1. The left singular vectors of ADH span the same space as the left singular vectors of A.
2. The matrix ADH has (up to log n factors) uniform leverage scores .

   (Thanks to V^THD having bounded entries – the proof closely follows JL-type proofs.)
3. We can approximate the left singular vectors of ADH (and thus the left singular vectors of A) by uniformly sampling columns of ADH.
4. The orthonormality of HD and a version of our relative-error Frobenius norm bound (involving approximately optimal sampling probabilities) suffice to show that (w.h.p.)

$$\left\| A - \tilde{C}\tilde{C}^{\dagger}A \right\|_F \leq (1+\epsilon)\left\| A - A_k \right\|_F$$

Uniform sample of $s = O\left(\frac{k}{\epsilon^2}\log^{c_0}\frac{n}{\epsilon}\right)$ columns of ADH



Running time

Let A by an m-by-n matrix whose SVD is:

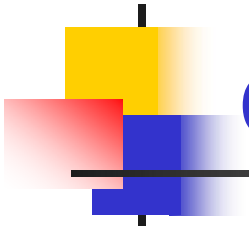
$$A = U\Sigma V^T \in \mathbb{R}^{m\times n}$$

Apply the (HD) part of the (PHD) transform to A.

$$ADH = U\Sigma \underbrace{\left(V^T DH\right)}_{\text{orthogonal matrix}} \in \mathbb{R}^{m\times n}$$

Running time:

1. Trivial analysis: first, uniformly sample s columns of DH and then compute their product with A.
   Takes O(mns) = O(mnk polylog(n)) time, already better than full SVD.
2. Less trivial analysis: take advantage of the fact that H is a Hadamard-Walsh matrix
   Improves the running time O(mn polylog(n) + mk^2polylog(n)).



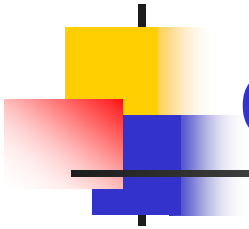
Conclusions

• Randomization and sampling can be used to solve problems that are massive and/or computationally expensive.

• By (carefully) sampling rows/columns/**entries** of a matrix, we can construct new sparse/smaller matrices that behave like the original matrix.

  • Can entry-wise sampling be made competitive to column-sampling in terms of accuracy and speed?
  See Achlioptas and McSherry (2001) STOC, (2007) JACM.
  • We recently improved/generalized/simplified it .
  See Nguyen, Drineas, & Tran (2010), Drineas & Zouzias (2010).
  • Exact reconstruction possible using uniform sampling for constant-rank matrices that satisfy certain (strong) assumptions.
  See Candes & Recht (2008), Candes & Tao (2009), Recht (2009).

• By preprocessing the matrix using random projections, we can sample rows/ columns much less carefully (even uniformly at random) and still get nice “behavior”.



Conclusions

• Randomization and sampling can be used to solve problems that are massive and/or computationally expensive.

• By (carefully) sampling rows/columns/**entries** of a matrix, we can construct new sparse/smaller matrices that behave like the original matrix.

  • Can entry-wise sampling be made competitive to column-sampling in terms of accuracy and speed?
  See Achlioptas and McSherry (2001) STOC, (2007) JACM.
  • We recently improved/generalized/simplified it .
  See Nguyen, Drineas, & Tran (2010), Drineas & Zouzias (2010).
  • Exact reconstruction possible using uniform sampling for constant-rank matrices that satisfy certain (strong) assumptions.
  See Candes & Recht (2008), Candes & Tao (2009), Recht (2009).

• By preprocessing the matrix using random projections, we can sample rows/ columns much less carefully (even uniformly at random) and still get nice “behavior”.