Randomized Algorithms for Very Large-Scale Linear Algebra

Gunnar Martinsson
The University of Colorado at Boulder

Collaborators: Edo Liberty, Vladimir Rokhlin, Yoel Shkolnisky, Arthur Szlam, Joel Tropp, Mark Tygert, Franco Woolfe, ...

Student: Nathan Halko (now at Spot Influence, LLC).

Postdoc: Sergey Voronin.

Survey: N. Halko, P.G. Martinsson, J. Tropp. *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions.* SIAM Review, **53**(2), 2011.

Slides: Posted under the "Talks" tab on my webpage (google "Gunnar Martinsson").

Support: NSF (#0748488, #0941476, #1320652), DARPA (N66001-13-1-4050).

Objective:

Given an $n \times n$ matrix **A**, for a very large n (say $n \sim 10^6$),
we seek to compute a rank-k approximation, with $k \ll n$ (say $k \sim 10^2$ or 10^3),

$$\underset{n \times n}{\mathbf{A}} \approx \underset{n \times k}{\mathbf{E}} \underset{k \times n}{\mathbf{F}^*} = \sum_{j=1}^{k} \mathbf{e}_j \, \mathbf{f}_j^*.$$

Solving this problem leads to algorithms for computing:

- Eigenvectors corresponding to leading eigenvalues.
  (Require $\mathbf{e}_j = \lambda_j \, \mathbf{f}_j$, and $\{\mathbf{f}_j\}_{j=1}^k$ to be orthonormal.)
- Singular Value Decomposition (SVD) / Principal Component Analysis (PCA).
  (Require $\{\mathbf{e}_j\}_{j=1}^k$ and $\{\mathbf{f}_j\}_{j=1}^k$ to be orthogonal sets.)
- Spanning columns or rows.
  (Require $\{\mathbf{e}_j\}_{j=1}^k$ to be columns of **A**, or require $\{\mathbf{f}_j^*\}_{j=1}^k$ to be rows of **A**.)
- *etc*

The problem being addressed is ubiquitous in applications.

Note: The only error that will be controlled is the "backwards error" $||\mathbf{A} - \mathbf{EF}|| \leq$ TOL.

Applications:

- Fast algorithms for elliptic PDEs: more efficient Fast Multipole Methods, fast *direct* solvers, construction of special quadratures for corners and edges, etc.

- Statistical analysis via Principal Component Analysis (PCA).

- Data mining (machine learning, analysis of network matrices, imaging, etc).

- Diffusion geometry; a technique for constructing parameterizations on large collections of data points organized (modulo noise) along non-linear low-dimensional manifolds. Requires the computations of eigenvectors of *graph Laplace operators*.

- Nearest neighbor search for large clouds of points in high dimensional space.

- General pre-conditioners.

- Etc.

Review of existing methods I

For a dense $n \times n$ matrix that fits in RAM, excellent algorithms are already part of LAPACK (and Matlab, *etc*).

- Double precision accuracy.
- Very stable.
- $O(n^3)$ asymptotic complexity. Reasonably small constants.
- Require extensive random access to the matrix.

When the target rank k is much smaller than n, there also exist $O(n^2\,k)$ methods with similar characteristics (the well-known Golub-Businger method, RRQR by Gu and Eisentstat, *etc*).

For small[a] matrices the state-of-the-art is very satisfactory.
For kicks, we will improve on it anyway, but this is not the main point.

[a] On a standard 2014 personal computer, "small" means $n \leq 10\,000$ or so.

Review of existing methods II

If the matrix is large, but can rapidly be applied to a vector (if it is sparse, or sparse in Fourier space, or amenable to the FMM, etc.), so called *Krylov subspace methods* often yield excellent accuracy and speed.

The idea is to pick a starting vector ω (often a random vector), "restrict" the matrix $\mathbf{A}$ to the k-dimensionsal "Krylov subspace"

$$\text{Span}(\omega,\, \mathbf{A}\,\omega,\, \mathbf{A}^2\,\omega,\, \ldots,\, \mathbf{A}^{k-1}\,\omega)$$

and compute approximate eigenvectors of the resulting matrix. Advantages:

- Very simple access to $\mathbf{A}$.
- Extremely high accuracy possible.

Drawbacks:

- In standard implementations, the matrix is revisited $O(k)$ times if a rank-k approximation is sought. (Blocked versions exist, but the convergence analysis is less developed.)
- There are numerical stability issues. These are well-studied and can be overcome, but they make the algorithms less portable (between applications, hardware platforms, etc.).

"New" challenges in algorithmic design:

The existing state-of-the-art methods of numerical linear algebra that we have very briefly outlined were designed for an environment where the matrix fits in RAM and the key to performance was to minimize the number of *floating point operations* required.

Currently, *communication* is becoming the real bottleneck:

- While clock speed is hardly improving at all anymore, the cost of a flop keeps going down rapidly. (Multi-core processors, GPUs, cloud computing, etc.)

- The cost of slow storage (hard drives, flash memory, etc.) is also going down rapidly.

- Communication costs are decreasing, but *not* rapidly.
  - Moving data from a hard-drive.
  - Moving data between nodes of a parallel machine. (Or cloud computer ...)
  - The amount of fast cache memory close to a processor is not improving much. (In fact, it could be said to be *shrinking* — GPUs, multi-core, etc.)

- "Deluge of data". Driven by ever cheaper storage and acquisition techniques. Web search, data mining in archives of documents or photos, hyper-spectral imagery, social networks, gene arrays, proteomics data, sensor networks, financial transactions, ...

The more powerful computing machinery becomes,
the more important efficient algorithm design becomes.

- Linear scaling (w.r.t. problem size, processors, etc.).
- Minimal data movement.

That *randomization* can be used to overcome some of the communication bottlenecks in matrix computations has been pointed out by several authors:

C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala (2000)

A. Frieze, R. Kannan, and S. Vempala (1999, 2004)

D. Achlioptas and F. McSherry (2001)

P. Drineas, R. Kannan, M. W. Mahoney, and S. Muthukrishnan (2006a, 2006b, 2006c, 2006d, *etc*)

S. Har-Peled (2006)

A. Deshpande and S. Vempala (2006)

S. Friedland, M. Kaveh, A. Niknejad, and H. Zare (2006)

T. Sarlós (2006a, 2006b, 2006c)

K. Clarkson, D. Woodruff (2009)

Literature survey: Halko, Martinsson, Tropp (2011).

Review of existing methods III

Examples of how randomization could be used:

- **Random column/row selection**
  Draw at random some columns and suppose that they span the entire column space.
  If rows are drawn as well, then spectral properties can be estimated.
  Crude sampling leads to less than $O(mn)$ complexity, but is very dangerous.

- **Sparsification**
  Zero out the vast majority of the entries of the matrix. Keep a random subset of entries, and boost their magnitude to preserve "something."

- **Quantization and sparsification**
  Restrict the entries of the matrix to a small set of values (-1/0/1 for instance).

The methods outlined can be as fast as you like, but must necessarily have very weak performance guarantees. They can work well for certain classes of matrices for which additional information is available (basically, matrices that are in some sense "over-sampled").

Approach advocated here:

A randomized algorithm for computing a rank-k approximation to an $m \times n$ matrix.
It is engineered from the ground up to:

- Minimize communication.
- Handle streaming data, or data stored "out-of-core."
- Easily adapt to a broad range of distributed computing architectures.

Computational profile:

- At least $O(mn)$ complexity (in fact $O(mnk)$ or $O(mn \log(k))$).
- The accuracy ε is a user-set number.
  (If the application permits, it could be $\varepsilon = 10^{-12}$ or less.)
- Since the method is randomized, it has a *failure probability* η.
  η is a user specified number.
  The cost of the method grows as $\eta \to 0$, but setting $\eta = 10^{-10}$ is cheap.
  For all practical purposes, the methods succeed with probability 1.

Goal: Given an $m \times n$ matrix $\mathbf{A}$, compute an approximate rank-k SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.

Goal: Given an $m \times n$ matrix $\mathbf{A}$, compute an approximate rank-k SVD $\mathbf{A} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$.

Algorithm:

1.
2.
3.

Find an $m \times k$ orthonormal matrix $\mathbf{Q}$ such that $\mathbf{A} \approx \mathbf{Q}\,\mathbf{Q}^*\,\mathbf{A}$.
(I.e., the columns of $\mathbf{Q}$ form an ON-basis for the range of $\mathbf{A}$.)

4. Form the $k \times n$ matrix $\mathbf{B} = \mathbf{Q}^*\,\mathbf{A}$.
5. Compute the SVD of the small matrix $\mathbf{B}$: $\mathbf{B} = \hat{\mathbf{U}}\,\mathbf{\Sigma}\,\mathbf{V}^*$.
6. Form the matrix $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$.

Goal: Given an $m \times n$ matrix $\mathbf{A}$, compute an approximate rank-k SVD $\mathbf{A} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$.

Algorithm:

1.
2. *Find an $m \times k$ orthonormal matrix $\mathbf{Q}$ such that $\mathbf{A} \approx \mathbf{Q}\,\mathbf{Q}^*\,\mathbf{A}$.*
   (I.e., the columns of $\mathbf{Q}$ form an ON-basis for the range of $\mathbf{A}$.)
3.
4. Form the $k \times n$ matrix $\mathbf{B} = \mathbf{Q}^*\,\mathbf{A}$.
5. Compute the SVD of the small matrix $\mathbf{B}$: $\mathbf{B} = \hat{\mathbf{U}}\,\mathbf{\Sigma}\,\mathbf{V}^*$.
6. Form the matrix $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$.

Note: Steps 4 – 6 are exact; the error in the method is all in $\mathbf{Q}$:

$$||\mathbf{A} - \underbrace{\mathbf{U}}_{=\mathbf{Q}\hat{\mathbf{U}}}\mathbf{\Sigma}\mathbf{V}^*|| = ||\mathbf{A} - \mathbf{Q}\underbrace{\hat{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^*}_{=\mathbf{B}}|| = ||\mathbf{A} - \mathbf{Q}\underbrace{\mathbf{B}}_{\mathbf{Q}^*\mathbf{A}}|| = ||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}||.$$

Note: The classical *Golub-Businger algorithm* follows this pattern. It finds $\mathbf{Q}$ in Step 3 via direct orthogonalization of the columns of $\mathbf{A}$ via, e.g., Gram-Schmidt.

Range finding problem: Given an $m \times n$ matrix $\mathbf{A}$ and an integer $k < \min(m, n)$, find an orthonormal $m \times k$ matrix $\mathbf{Q}$ such that $\mathbf{A} \approx \mathbf{Q}\mathbf{Q}^*\mathbf{A}$.

Solving the primitive problem via randomized sampling — intuition:

1. Draw random vectors $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_k \in \mathbb{R}^n$.
   (We will discuss the choice of distribution later — think Gaussian for now.)

2. Form "sample" vectors $\mathbf{y}_1 = \mathbf{A}\,\boldsymbol{\omega}_1, \mathbf{y}_2 = \mathbf{A}\,\boldsymbol{\omega}_2, \ldots, \mathbf{y}_k = \mathbf{A}\,\boldsymbol{\omega}_k \in \mathbb{R}^m$.

3. Form orthonormal vectors $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k \in \mathbb{R}^m$ such that

$$\text{Span}(\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k) = \text{Span}(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k).$$

   For instance, Gram-Schmidt can be used — pivoting is rarely required.

If $\mathbf{A}$ has *exact* rank k, then $\text{Span}\{\mathbf{q}_j\}_{j=1}^k = \text{Ran}(\mathbf{A})$ with probability 1.

Range finding problem: Given an $m \times n$ matrix $\mathbf{A}$ and an integer $k < \min(m, n)$, find an orthonormal $m \times k$ matrix $\mathbf{Q}$ such that $\mathbf{A} \approx \mathbf{Q}\mathbf{Q}^*\mathbf{A}$.

Solving the primitive problem via randomized sampling — intuition:

1. Draw random vectors $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_k \in \mathbb{R}^n$.
   (We will discuss the choice of distribution later — think Gaussian for now.)

2. Form "sample" vectors $\mathbf{y}_1 = \mathbf{A}\,\boldsymbol{\omega}_1, \mathbf{y}_2 = \mathbf{A}\,\boldsymbol{\omega}_2, \ldots, \mathbf{y}_k = \mathbf{A}\,\boldsymbol{\omega}_k \in \mathbb{R}^m$.

3. Form orthonormal vectors $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k \in \mathbb{R}^m$ such that

$$\text{Span}(\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k) = \text{Span}(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k).$$

   For instance, Gram-Schmidt can be used — pivoting is rarely required.

If $\mathbf{A}$ has *exact* rank k, then $\text{Span}\{\mathbf{q}_j\}_{j=1}^k = \text{Ran}(\mathbf{A})$ with probability 1.

What is perhaps surprising is that even in the general case, $\{\mathbf{q}_j\}_{j=1}^k$ often does almost as good of a job as the theoretically optimal vectors (which happen to be the k leading left singular vectors).

Range finding problem: Given an $m \times n$ matrix **A** and an integer $k < \min(m, n)$, find an orthonormal $m \times k$ matrix **Q** such that $\mathbf{A} \approx \mathbf{Q}\mathbf{Q}^*\mathbf{A}$.

Solving the primitive problem via randomized sampling — intuition:

1. Draw a random matrix $\mathbf{\Omega} \in \mathbb{R}^{n \times k}$.
   (We will discuss the choice of distribution later — think Gaussian for now.)

2. Form a "sample" matrix $\mathbf{Y} = \mathbf{A}\,\mathbf{\Omega} \in \mathbb{R}^{m \times k}$.

3. Form an orthonormal matrix $\mathbf{Q} \in \mathbb{R}^{m \times k}$ such that $\mathbf{Y} = \mathbf{Q}\mathbf{R}$.
   For instance, Gram-Schmidt can be used — pivoting is rarely required.

If **A** has *exact* rank k, then $\mathbf{A} = \mathbf{Q}\mathbf{Q}^*\mathbf{A}$ with probability 1.

Goal: Given an $m \times n$ matrix $\mathbf{A}$, compute an approximate rank-k SVD $\mathbf{A} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$.

Algorithm:

1. Draw an $n \times k$ Gaussian random matrix $\mathbf{\Omega}$.
2. Form the $m \times k$ sample matrix $\mathbf{Y} = \mathbf{A}\,\mathbf{\Omega}$.
3. Form an $m \times k$ orthonormal matrix $\mathbf{Q}$ such that $\mathbf{Y} = \mathbf{Q}\,\mathbf{R}$.
4. Form the $k \times n$ matrix $\mathbf{B} = \mathbf{Q}^*\,\mathbf{A}$.
5. Compute the SVD of the small matrix $\mathbf{B}$: $\mathbf{B} = \hat{\mathbf{U}}\,\mathbf{\Sigma}\,\mathbf{V}^*$.
6. Form the matrix $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$.

Goal: Given an $m \times n$ matrix $\mathbf{A}$, compute an approximate rank-k SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.

Algorithm:

1. Draw an $n \times k$ Gaussian random matrix $\mathbf{\Omega}$. `Omega = randn(n,k)`
2. Form the $m \times k$ sample matrix $\mathbf{Y} = \mathbf{A\,\Omega}$. `Y = A * Omega`
3. Form an $m \times k$ orthonormal matrix $\mathbf{Q}$ such that $\mathbf{Y} = \mathbf{Q\,R}$. `[Q, R] = qr(Y)`
4. Form the $k \times n$ matrix $\mathbf{B} = \mathbf{Q}^*\,\mathbf{A}$. `B = Q' * A`
5. Compute the SVD of the small matrix $\mathbf{B}$: $\mathbf{B} = \hat{\mathbf{U}}\,\mathbf{\Sigma}\,\mathbf{V}^*$. `[Uhat, Sigma, V] = svd(B,0)`
6. Form the matrix $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$. `U = Q * Uhat`

Single pass algorithms:

A is symmetric:
Generate a random matrix $\mathbf{\Omega}$.
Compute a sample matrix $\mathbf{Y}$.
Find an ON matrix $\mathbf{Q}$ such that $\mathbf{Y} = \mathbf{Q}\,\mathbf{Q}^*\,\mathbf{Y}$.
Solve for $\mathbf{T}$ the linear system $\mathbf{Q}^*\,\mathbf{Y} = \mathbf{T}\,(\mathbf{Q}^*\,\mathbf{\Omega})$.
Factor $\mathbf{T}$ so that $\mathbf{T} = \hat{\mathbf{U}}\,\mathbf{\Lambda}\,\hat{\mathbf{U}}^*$.
Form $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$.
Output: $\mathbf{A} \approx \mathbf{U}\mathbf{\Lambda}\mathbf{U}^*$

Note: With $\mathbf{B}$ as on the previous slide we have $\mathbf{T} \approx \mathbf{B}\,\mathbf{Q}$ (sym. case) and $\mathbf{T} \approx \mathbf{B}\,\mathbf{W}$ (nonsym. case).

Note: An alternative procedure based on "interpolative decompositions" is often more accurate.

References: *Woolfe, Liberty, Rokhlin, and Tygert (2008), Clarkson and Woodruff (2009), HMT2009, HMT2011.*

Single pass algorithms:

A is symmetric:	**A is not symmetric:**
Generate a random matrix $\mathbf{\Omega}$.	Generate random matrices $\mathbf{\Omega}$ and $\mathbf{\Psi}$.
Compute a sample matrix $\mathbf{Y}$.	Compute sample matrices $\mathbf{Y} = \mathbf{A}\,\mathbf{\Omega}$ and $\mathbf{Z} = \mathbf{A}^*\,\mathbf{\Psi}$.
Find an ON matrix $\mathbf{Q}$ such that $\mathbf{Y} = \mathbf{Q}\,\mathbf{Q}^*\,\mathbf{Y}$.	Find ON matrices $\mathbf{Q}$ and $\mathbf{W}$ such that $\mathbf{Y} = \mathbf{Q}\,\mathbf{Q}^*\,\mathbf{Y}$ and $\mathbf{Z} = \mathbf{W}\,\mathbf{W}^*\,\mathbf{Z}$.
Solve for $\mathbf{T}$ the linear system $\mathbf{Q}^*\,\mathbf{Y} = \mathbf{T}\,(\mathbf{Q}^*\,\mathbf{\Omega})$.	Solve for $\mathbf{T}$ the linear systems $\mathbf{Q}^*\,\mathbf{Y} = \mathbf{T}\,(\mathbf{W}^*\,\mathbf{\Omega})$ and $\mathbf{W}^*\,\mathbf{Z} = \mathbf{T}^*\,(\mathbf{Q}^*\,\mathbf{\Psi})$.
Factor $\mathbf{T}$ so that $\mathbf{T} = \hat{\mathbf{U}}\,\mathbf{\Lambda}\,\hat{\mathbf{U}}^*$.	Factor $\mathbf{T}$ so that $\mathbf{T} = \hat{\mathbf{U}}\,\mathbf{\Sigma}\,\hat{\mathbf{V}}^*$.
Form $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$.	Form $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$ and $\mathbf{V} = \mathbf{W}\,\hat{\mathbf{V}}$.
Output: $\mathbf{A} \approx \mathbf{U}\mathbf{\Lambda}\mathbf{U}^*$	Output: $\mathbf{A} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$

Note: With $\mathbf{B}$ as on the previous slide we have $\mathbf{T} \approx \mathbf{B}\,\mathbf{Q}$ (sym. case) and $\mathbf{T} \approx \mathbf{B}\,\mathbf{W}$ (nonsym. case).

Note: An alternative procedure based on "interpolative decompositions" is often more accurate.

References: *Woolfe, Liberty, Rokhlin, and Tygert (2008), Clarkson and Woodruff (2009), HMT2009, HMT2011.*

Cost of randomized methods vs. classical (deterministic) methods:

Case 1 — A is given as an array of numbers that fits in RAM ("small matrix"):

Classical methods (e.g. Golub-Businger) have cost $O(mnk)$. The basic randomized method described also has $O(mnk)$ cost, but with a lower pre-factor (and sometimes lower accuracy). However, the cost can be reduced to $O(mn\log(k))$ if a structured random matrix is used. For instance, $\mathbf{\Omega}$ can be a sub-sampled randomized Fourier transform, which can be applied rapidly using variations of the FFT.

Example of a *subsampled random Fourier Transform (SRFT)*

$$\begin{array}{ccccc} \mathbf{\Omega} & = & \mathbf{D} & \mathbf{F} & \mathbf{S} \\ n \times k & & n \times n & n \times n & n \times k \end{array}$$

- **D** is a diagonal matrix whose entries are i.i.d. random variables drawn from a uniform distribution on the unit circle in $\mathbb{C}$.
- **F** is the discrete Fourier transform, $\mathbf{F}_{pq} = \frac{1}{\sqrt{n}} e^{-2\pi i(p-1)(q-1)/n}$.
- **S** is a matrix whose entries are all zeros except for a single, randomly placed 1 in each column. (In other words, the action of **S** is to draw k columns at random from **D F**.)

References: Ailon and Chazelle (2006); Liberty, Rokhlin, Tygert, and Woolfe (2006).

To achieve overall complexity $O(mn\log(k))$, the scheme should be slightly modified so that randomized sampling is used to determine both the column space and the row space:

Given an $m \times n$ matrix $\mathbf{A}$, find $\mathbf{U}$, $\mathbf{V}$, $\mathbf{\Sigma}$ of rank k such that $\mathbf{A} \approx \mathbf{U}\,\mathbf{\Sigma}\,\mathbf{V}^*$.

1. Choose an over-sampling parameter p and set $\ell = k + p$. (Say $p = k$ so $\ell = 2k$.)

2. Generate SRFT's $\mathbf{\Omega}$ and $\mathbf{\Psi}$ of sizes $n \times \ell$, and $m \times \ell$.

3. Form the sample matrices $\mathbf{Y} = \mathbf{A}\,\mathbf{\Omega}$ and $\mathbf{Z} = \mathbf{A}^*\,\mathbf{\Psi}$.

4. Find ON matrices $\mathbf{Q}$ and $\mathbf{W}$ such that $\mathbf{Y} = \mathbf{Q}\,\mathbf{Q}^*\,\mathbf{Y}$ and $\mathbf{Z} = \mathbf{W}\,\mathbf{W}^*\,\mathbf{Z}$.

5. Solve for the $k \times k$ matrix $\mathbf{T}$ the systems $\mathbf{Q}^*\,\mathbf{Y} = \mathbf{T}\,(\mathbf{W}^*\,\mathbf{\Omega})$ and $\mathbf{W}^*\,\mathbf{Z} = \mathbf{T}^*\,(\mathbf{Q}^*\,\mathbf{\Psi})$.

6. Compute the SVD of the small matrix $\mathbf{T} = \hat{\mathbf{U}}\,\mathbf{\Sigma}\,\hat{\mathbf{V}}^*$ (and truncate if desired).

7. Form $\mathbf{U} = \mathbf{Q}\,\hat{\mathbf{U}}$ and $\mathbf{V} = \mathbf{W}\,\hat{\mathbf{V}}$.

Observation 1: Forming $\mathbf{A}\,\mathbf{\Omega}$ and $\mathbf{A}^*\,\mathbf{\Psi}$ in Step 2 has cost $O(m\,n\,\log(k))$ since $\ell \sim k$.

Observation 2: All other steps cost at most $O((m+n)\,k^2)$.

Practical speed of $O(mnk\log(k))$ complexity randomized

Consider the task of computing a rank-k SVD of a matrix **A** of size $n \times n$.

$t^{(\text{direct})}$ Time for classical (Golub-Businger) method — $O(k\,n^2)$

$t^{(\text{srft})}$ Time for randomized method with an SRFT — $O(\log(k)\,n^2)$

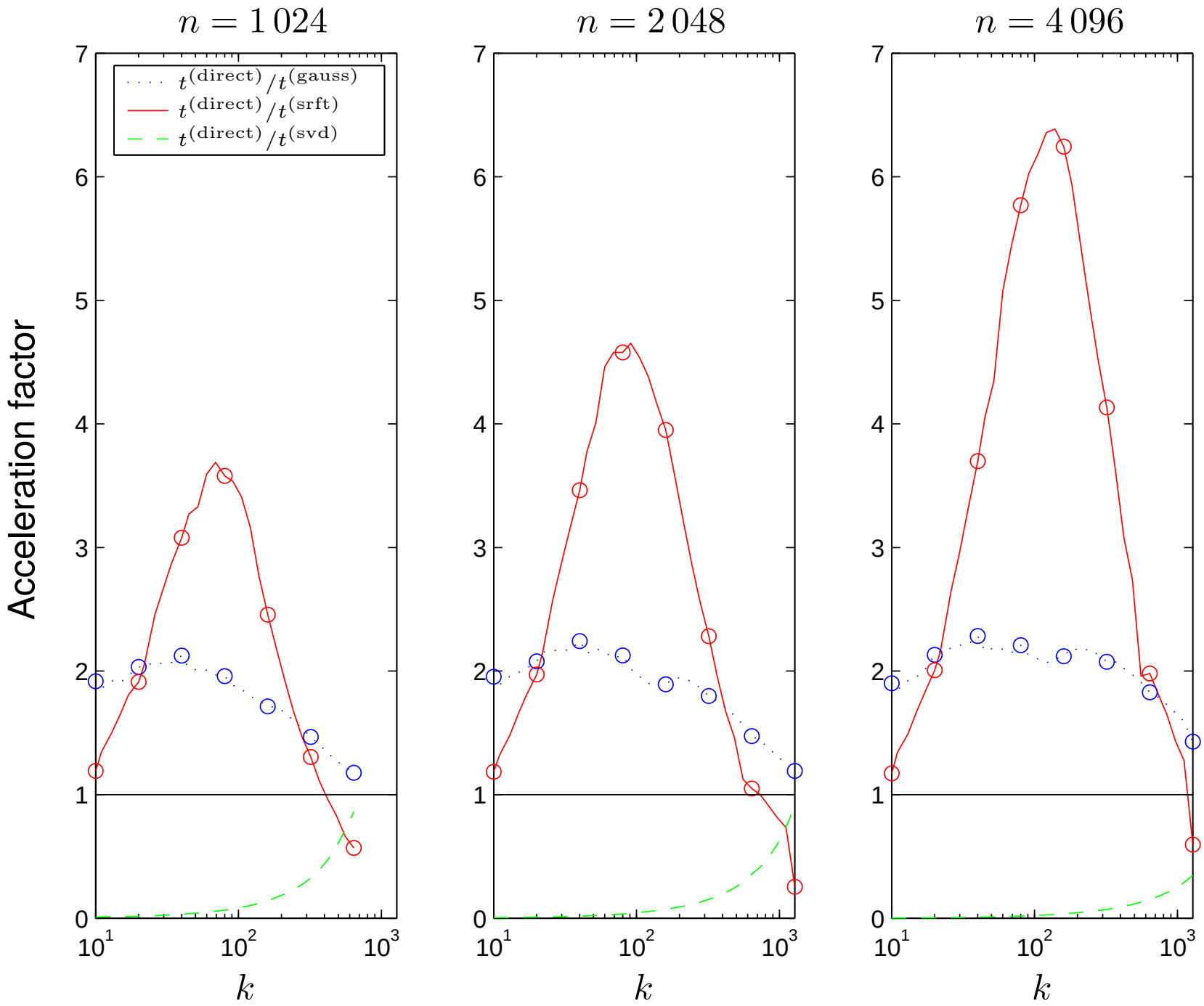
$t^{(\text{gauss})}$ Time for randomized method with a Gaussian matrix — $O(k\,n^2)$

$t^{(\text{svd})}$ Time for a full SVD — $O(n^3)$

We will show the

acceleration factors: $\dfrac{t^{(\text{direct})}}{t^{(\text{srft})}}$ $\dfrac{t^{(\text{direct})}}{t^{(\text{gauss})}}$ $\dfrac{t^{(\text{direct})}}{t^{(\text{svd})}}$

for different values of n and k.



Observe: Large speedups (up to a factor 6!) for moderate size matrices.

Cost of randomized methods vs. classical (deterministic) methods:

Case 1 — A is given as an array of numbers that fits in RAM ("small matrix"):

Classical methods (e.g. Golub-Businger) have cost $O(mnk)$. The basic randomized method described also has $O(mnk)$ cost, but with a lower pre-factor (and sometimes lower accuracy). However, the cost can be reduced to $O(mn\log(k))$ if a structured random matrix is used. For instance, $\mathbf{\Omega}$ can be a sub-sampled randomized Fourier transform, which can be applied rapidly using variations of the FFT.

Cost of randomized methods vs. classical (deterministic) methods:

Case 1 — A is given as an array of numbers that fits in RAM ("small matrix"):

Classical methods (e.g. Golub-Businger) have cost $O(mnk)$. The basic randomized method described also has $O(mnk)$ cost, but with a lower pre-factor (and sometimes lower accuracy). However, the cost can be reduced to $O(mn\log(k))$ if a structured random matrix is used. For instance, $\mathbf{\Omega}$ can be a sub-sampled randomized Fourier transform, which can be applied rapidly using variations of the FFT.

Case 2 — A is given as an array of numbers on disk ("large matrix"):

In this case, the relevant metric is memory access. Randomized methods access **A** via sweeps over the entire matrix. With slight modifications, the randomized method can be executed in a *single pass* over the matrix. High accuracy can be attained with a small number of passes (say two, or five).

(In contrast, classical (deterministic) methods require "random" access to matrix elements...)

COST OF RANDOMIZED METHODS VS. CLASSICAL (DETERMINISTIC) METHODS:

Case 1 — A is given as an array of numbers that fits in RAM ("small matrix"):

Classical methods (e.g. Golub-Businger) have cost $O(mnk)$. The basic randomized method described also has $O(mnk)$ cost, but with a lower pre-factor (and sometimes lower accuracy). However, the cost can be reduced to $O(mn\log(k))$ if a structured random matrix is used. For instance, $\mathbf{\Omega}$ can be a sub-sampled randomized Fourier transform, which can be applied rapidly using variations of the FFT.

Case 2 — A is given as an array of numbers on disk ("large matrix"):

In this case, the relevant metric is memory access. Randomized methods access **A** via sweeps over the entire matrix. With slight modifications, the randomized method can be executed in a *single pass* over the matrix. High accuracy can be attained with a small number of passes (say two, or five).

(In contrast, classical (deterministic) methods require "random" access to matrix elements...)

Case 3 — A and A^* can be applied fast ("structured matrix"):

Think of **A** sparse, or sparse in the Fourier domain, or amenable to the Fast Multipole Method, etc. The classical competitor is in this case "Krylov methods". Randomized methods tend to be more robust, and easier to implement in massively parallel environments. They are more easily blocked to reduce communication. However, Krylov methods sometimes lead to higher accuracy.

Theory

Input: An $m \times n$ matrix $\mathbf{A}$ and a target rank k.

Output: Rank-k factors $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{V}$ in an approximate SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.

(1) Draw an $n \times k$ random matrix $\mathbf{\Omega}$.	(4) Form the small matrix $\mathbf{B} = \mathbf{Q}^* \mathbf{A}$.
(2) Form the $n \times k$ sample matrix $\mathbf{Y} = \mathbf{A\Omega}$.	(5) Factor the small matrix $\mathbf{B} = \hat{\mathbf{U}}\mathbf{\Sigma V}^*$.
(3) Compute an ON matrix $\mathbf{Q}$ s.t. $\mathbf{Y} = \mathbf{QQ}^*\mathbf{Y}$.	(6) Form $\mathbf{U} = \mathbf{Q}\hat{\mathbf{U}}$.

Question: What is the error $e_k = ||\mathbf{A} - \mathbf{U\Sigma V}^*||$? (Recall $e_k = ||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}||$.)

Input: An $m \times n$ matrix **A** and a target rank k.

Output: Rank-k factors **U**, **Σ**, and **V** in an approximate SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.

(1) Draw an $n \times k$ random matrix $\mathbf{\Omega}$.	(4) Form the small matrix $\mathbf{B} = \mathbf{Q}^* \mathbf{A}$.
(2) Form the $n \times k$ sample matrix $\mathbf{Y} = \mathbf{A\Omega}$.	(5) Factor the small matrix $\mathbf{B} = \hat{\mathbf{U}}\mathbf{\Sigma V}^*$.
(3) Compute an ON matrix **Q** s.t. $\mathbf{Y} = \mathbf{QQ}^*\mathbf{Y}$.	(6) Form $\mathbf{U} = \mathbf{Q}\hat{\mathbf{U}}$.

Question: What is the error $e_k = ||\mathbf{A} - \mathbf{U\Sigma V}^*||$? (Recall $e_k = ||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}||$.)

Eckart-Young theorem: e_k is bounded from below by the $(k+1)$'th singular value σ_{k+1} of **A**.

Question: Is e_k close to σ_{k+1}?

Input: An $m \times n$ matrix **A** and a target rank k.	
Output: Rank-k factors **U**, **Σ**, and **V** in an approximate SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.	
(1) Draw an $n \times k$ random matrix **Ω**. (2) Form the $n \times k$ sample matrix $\mathbf{Y} = \mathbf{A\Omega}$. (3) Compute an ON matrix **Q** s.t. $\mathbf{Y} = \mathbf{QQ}^*\mathbf{Y}$.	(4) Form the small matrix $\mathbf{B} = \mathbf{Q}^* \mathbf{A}$. (5) Factor the small matrix $\mathbf{B} = \hat{\mathbf{U}}\mathbf{\Sigma V}^*$. (6) Form $\mathbf{U} = \mathbf{Q}\hat{\mathbf{U}}$.

Question: What is the error $e_k = ||\mathbf{A} - \mathbf{U\Sigma V}^*||$? (Recall $e_k = ||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}||$.)

Eckart-Young theorem: e_k is bounded from below by the $(k+1)$'th singular value σ_{k+1} of **A**.

Question: Is e_k close to σ_{k+1}?

Answer: Lamentably, no. The expectation of $\frac{e_k}{\sigma_{k+1}}$ is large, and has very large variance.

Input: An $m \times n$ matrix **A** and a target rank k.	
Output: Rank-k factors **U**, **Σ**, and **V** in an approximate SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.	
(1) Draw an $n \times k$ random matrix **Ω**.	(4) Form the small matrix $\mathbf{B} = \mathbf{Q}^* \mathbf{A}$.
(2) Form the $n \times k$ sample matrix $\mathbf{Y} = \mathbf{A\Omega}$.	(5) Factor the small matrix $\mathbf{B} = \hat{\mathbf{U}}\mathbf{\Sigma V}^*$.
(3) Compute an ON matrix **Q** s.t. $\mathbf{Y} = \mathbf{QQ}^*\mathbf{Y}$.	(6) Form $\mathbf{U} = \mathbf{Q}\hat{\mathbf{U}}$.

Question: What is the error $e_k = ||\mathbf{A} - \mathbf{U\Sigma V}^*||$? (Recall $e_k = ||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}||$.)

Eckart-Young theorem: e_k is bounded from below by the $(k+1)$'th singular value σ_{k+1} of **A**.

Question: Is e_k close to σ_{k+1}?

Answer: Lamentably, no. The expectation of $\frac{e_k}{\sigma_{k+1}}$ is large, and has very large variance.

Remedy: Over-sample *slightly*. Compute $k+p$ samples from the range of **A**.
It turns out that $p = 5$ or 10 is often sufficient. $p = k$ is almost always more than enough.

Input: An $m \times n$ matrix **A**, a target rank k, *and an over-sampling parameter p (say $p = 5$).*	
Output: Rank-$(k+p)$ factors **U**, **Σ**, and **V** in an approximate SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.	
(1) Draw an $n \times (k+p)$ random matrix **Ω**.	(4) Form the small matrix $\mathbf{B} = \mathbf{Q}^* \mathbf{A}$.
(2) Form the $n \times (k+p)$ sample matrix $\mathbf{Y} = \mathbf{A\Omega}$.	(5) Factor the small matrix $\mathbf{B} = \hat{\mathbf{U}}\mathbf{\Sigma V}^*$.
(3) Compute an ON matrix **Q** s.t. $\mathbf{Y} = \mathbf{QQ}^*\mathbf{Y}$.	(6) Form $\mathbf{U} = \mathbf{Q}\hat{\mathbf{U}}$.

Bound on the expectation of the error for Gaussian test matrices

Let **A** denote an $m \times n$ matrix with singular values $\{\sigma_j\}_{j=1}^{\min(m,n)}$.

Let k denote a target rank and let p denote an over-sampling parameter.

Let **Ω** denote an $n \times (k+p)$ Gaussian matrix.

Let **Q** denote the $m \times (k+p)$ matrix $\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega})$.

If $p \geq 2$, then

$$\mathbb{E}||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}||_{\text{Frob}} \leq \left(1 + \frac{k}{p-1}\right)^{1/2} \left(\sum_{j=k+1}^{\min(m,n)} \sigma_j^2\right)^{1/2},$$

and

$$\mathbb{E}||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|| \leq \left(1 + \sqrt{\frac{k}{p-1}}\right) \sigma_{k+1} + \frac{e\sqrt{k+p}}{p} \left(\sum_{j=k+1}^{\min(m,n)} \sigma_j^2\right)^{1/2}.$$

Ref: Halko, Martinsson, Tropp, 2009 & 2011

Large deviation bound for the error for Gaussian test matrices

Let **A** denote an $m \times n$ matrix with singular values $\{\sigma_j\}_{j=1}^{\min(m,n)}$.

Let k denote a target rank and let p denote an over-sampling parameter.

Let **Ω** denote an $n \times (k+p)$ Gaussian matrix.

Let **Q** denote the $m \times (k+p)$ matrix $\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega})$.

If $p \geq 4$, and u and t are such that $u \geq 1$ and $t \geq 1$, then

$$||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|| \leq \left(1 + t\sqrt{\frac{3k}{p+1}} + u\,t\,\frac{e\sqrt{k+p}}{p+1}\right)\sigma_{k+1} + \frac{t\,e\sqrt{k+p}}{p+1}\left(\sum_{j>k}\sigma_j^2\right)^{1/2}$$

except with probability at most $2\,t^{-p} + e^{-u^2/2}$.

Ref: Halko, Martinsson, Tropp, 2009 & 2011; Martinsson, Rokhlin, Tygert (2006)

---

u and t parameterize "bad" events — large u, t is bad, but unlikely.

Certain choices of t and u lead to simpler results. For instance,

$$||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|| \leq \left(1 + 16\sqrt{1 + \frac{k}{p+1}}\right)\sigma_{k+1} + 8\frac{\sqrt{k+p}}{p+1}\left(\sum_{j>k}\sigma_j^2\right)^{1/2},$$

except with probability at most $3\,e^{-p}$.

Large deviation bound for the error for Gaussian test matrices

Let **A** denote an $m \times n$ matrix with singular values $\{\sigma_j\}_{j=1}^{\min(m,n)}$.

Let k denote a target rank and let p denote an over-sampling parameter.

Let **Ω** denote an $n \times (k+p)$ Gaussian matrix.

Let **Q** denote the $m \times (k+p)$ matrix $\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega})$.

If $p \geq 4$, and u and t are such that $u \geq 1$ and $t \geq 1$, then

$$||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|| \leq \left(1 + t\sqrt{\frac{3k}{p+1}} + u\,t\,\frac{e\sqrt{k+p}}{p+1}\right)\sigma_{k+1} + \frac{t\,e\sqrt{k+p}}{p+1}\left(\sum_{j>k}\sigma_j^2\right)^{1/2}$$

except with probability at most $2\,t^{-p} + e^{-u^2/2}$.

Ref: Halko, Martinsson, Tropp, 2009 & 2011; Martinsson, Rokhlin, Tygert (2006)

---

u and t parameterize "bad" events — large u, t is bad, but unlikely.

Certain choices of t and u lead to simpler results. For instance,

$$||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|| \leq \left(1 + 6\sqrt{(k+p)\cdot p\log p}\right)\sigma_{k+1} + 3\sqrt{k+p}\left(\sum_{j>k}\sigma_j^2\right)^{1/2},$$

except with probability at most $3\,p^{-p}$.

Proofs — Overview:

Let **A** denote an $m \times n$ matrix with singular values $\{\sigma_j\}_{j=1}^{\min(m,n)}$.

Let k denote a target rank and let p denote an over-sampling parameter. Set $\ell = k + p$.

Let $\mathbf{\Omega}$ denote an $n \times \ell$ "test matrix", and let **Q** denote the $m \times \ell$ matrix $\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega})$.

We seek to bound the error $e_k = e_k(\mathbf{A}, \mathbf{\Omega}) = ||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}||$, which is a random variable.

1. Make no assumption on $\mathbf{\Omega}$. Construct a deterministic bound of the form

$$||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|| \leq \cdots \mathbf{A} \cdots \mathbf{\Omega} \cdots$$

2. Assume that $\mathbf{\Omega}$ is drawn from a normal Gaussian distribution.
   Take expectations of the deterministic bound to attain a bound of the form

$$\mathbb{E}\big[||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}||\big] \leq \cdots \mathbf{A} \cdots$$

3. Assume that $\mathbf{\Omega}$ is drawn from a normal Gaussian distribution.
   Take expectations of the deterministic bound conditioned on "bad behavior" in $\mathbf{\Omega}$ to get

$$||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|| \leq \cdots \mathbf{A} \cdots$$

   that hold with probability at least $\cdots$.

Part 1 (out of 3) — deterministic bound:

Let **A** denote an $m \times n$ matrix with singular values $\{\sigma_j\}_{j=1}^{\min(m,n)}$.

Let k denote a target rank and let p denote an over-sampling parameter. Set $\ell = k + p$.

Let $\mathbf{\Omega}$ denote an $n \times \ell$ "test matrix", and let **Q** denote the $m \times \ell$ matrix $\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega})$.

Partition the SVD of **A** as follows:

$$\mathbf{A} = \mathbf{U} \begin{array}{c} \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{bmatrix} \mathbf{\Sigma}_1 & \\ & \mathbf{\Sigma}_2 \end{bmatrix} \end{array} \begin{array}{c} \begin{array}{c} k \end{array} \\ \begin{bmatrix} \mathbf{V}_1^* \\ \mathbf{V}_2^* \end{bmatrix} \\ \begin{array}{c} n-k \end{array} \end{array}$$

Define $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ via

$$\mathbf{\Omega}_1 = \mathbf{V}_1^* \, \mathbf{\Omega} \quad \text{and} \quad \mathbf{\Omega}_2 = \mathbf{V}_2^* \, \mathbf{\Omega}.$$

Theorem: [HMT2009,HMT2011] Assuming that $\mathbf{\Omega}_1$ is not singular, it holds that

$$|||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}|||^2 \leq \underbrace{|||\mathbf{\Sigma}_2|||^2}_{\text{theoretically minimal error}} + |||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger|||^2.$$

Here, $|||\cdot|||$ represents either ℓ^2-operator norm, or the Frobenius norm.

Note: A similar (but weaker) result appears in Boutsidis, Mahoney, Drineas (2009).

Recall: $\mathbf{A} = \mathbf{U}\begin{bmatrix} \mathbf{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_2 \end{bmatrix}\begin{bmatrix} \mathbf{V}_1^* \\ \mathbf{V}_2^* \end{bmatrix}, \begin{bmatrix} \mathbf{\Omega}_1 \\ \mathbf{\Omega}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{V}_1^*\mathbf{\Omega} \\ \mathbf{V}_2^*\mathbf{\Omega} \end{bmatrix}$, $\mathbf{Y} = \mathbf{A\Omega}$, $\mathbf{P}$ proj$^{\text{n}}$ onto Ran$(\mathbf{Y})$.

Thm: Suppose $\mathbf{\Sigma}_1\mathbf{\Omega}_1$ has full rank. Then $||\mathbf{A} - \mathbf{PA}||^2 \leq ||\mathbf{\Sigma}_2||^2 + ||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger||^2$.

Proof: The problem is rotationally invariant $\Rightarrow$ We can assume $\mathbf{U} = \mathbf{I}$ and so $\mathbf{A} = \mathbf{\Sigma V}^*$.

Simple calculation: $||(\mathbf{I} - \mathbf{P})\mathbf{A}||^2 = ||\mathbf{A}^*(\mathbf{I} - \mathbf{P})^2\mathbf{A}|| = ||\mathbf{\Sigma}(\mathbf{I} - \mathbf{P})\mathbf{\Sigma}||$.

$$\text{Ran}(\mathbf{Y}) = \text{Ran}\left(\begin{bmatrix} \mathbf{\Sigma}_1\mathbf{\Omega}_1 \\ \mathbf{\Sigma}_2\mathbf{\Omega}_2 \end{bmatrix}\right) = \text{Ran}\left(\begin{bmatrix} \mathbf{I} \\ \mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1 \end{bmatrix}\mathbf{\Sigma}_1\mathbf{\Omega}_1\right) = \text{Ran}\left(\begin{bmatrix} \mathbf{I} \\ \mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1 \end{bmatrix}\right)$$

Set $\mathbf{F} = \mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1^{-1}$. Then $\mathbf{P} = \begin{bmatrix} \mathbf{I} \\ \mathbf{F} \end{bmatrix}(\mathbf{I} + \mathbf{F}^*\mathbf{F})^{-1}[\mathbf{I}\ \mathbf{F}^*]$. (Compare to $\mathbf{P}_{\text{ideal}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$.)

Use properties of psd matrices: $\mathbf{I} - \mathbf{P} \preccurlyeq \cdots \preccurlyeq \begin{bmatrix} \mathbf{F}^*\mathbf{F} & -(\mathbf{I} + \mathbf{F}^*\mathbf{F})^{-1}\mathbf{F}^* \\ -\mathbf{F}(\mathbf{I} + \mathbf{F}^*\mathbf{F})^{-1} & \mathbf{I} \end{bmatrix}$

Conjugate by $\mathbf{\Sigma}$ to get $\mathbf{\Sigma}(\mathbf{I} - \mathbf{P})\mathbf{\Sigma} \preccurlyeq \begin{bmatrix} \mathbf{\Sigma}_1\mathbf{F}^*\mathbf{F}\mathbf{\Sigma}_1 & -\mathbf{\Sigma}_1(\mathbf{I} + \mathbf{F}^*\mathbf{F})^{-1}\mathbf{F}^*\mathbf{\Sigma}_2 \\ -\mathbf{\Sigma}_2\mathbf{F}(\mathbf{I} + \mathbf{F}^*\mathbf{F})^{-1}\mathbf{\Sigma}_1 & \mathbf{\Sigma}_2^2 \end{bmatrix}$

Diagonal dominance: $||\mathbf{\Sigma}(\mathbf{I} - \mathbf{P})\mathbf{\Sigma}|| \leq ||\mathbf{\Sigma}_1\mathbf{F}^*\mathbf{F}\mathbf{\Sigma}_1|| + ||\mathbf{\Sigma}_2^2|| = ||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger||^2 + ||\mathbf{\Sigma}_2||^2$.

Part 2 (out of 3) — bound on expectation of error when Ω is Gaussian:

Let **A** denote an $m \times n$ matrix with singular values $\{\sigma_j\}_{j=1}^{\min(m,n)}$.

Let k denote a target rank and let p denote an over-sampling parameter. Set $\ell = k + p$.

Let $\mathbf{\Omega}$ denote an $n \times \ell$ "test matrix", and let **Q** denote the $m \times \ell$ matrix $\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega})$.

Recall: $|||\mathbf{A} - \mathbf{QQ}^*\mathbf{A}|||^2 \leq |||\mathbf{\Sigma}_2|||^2 + |||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger|||^2$, where $\mathbf{\Omega}_1 = \mathbf{V}_1^*\,\mathbf{\Omega}$ and $\mathbf{\Omega}_2 = \mathbf{V}_2^*\,\mathbf{\Omega}$.

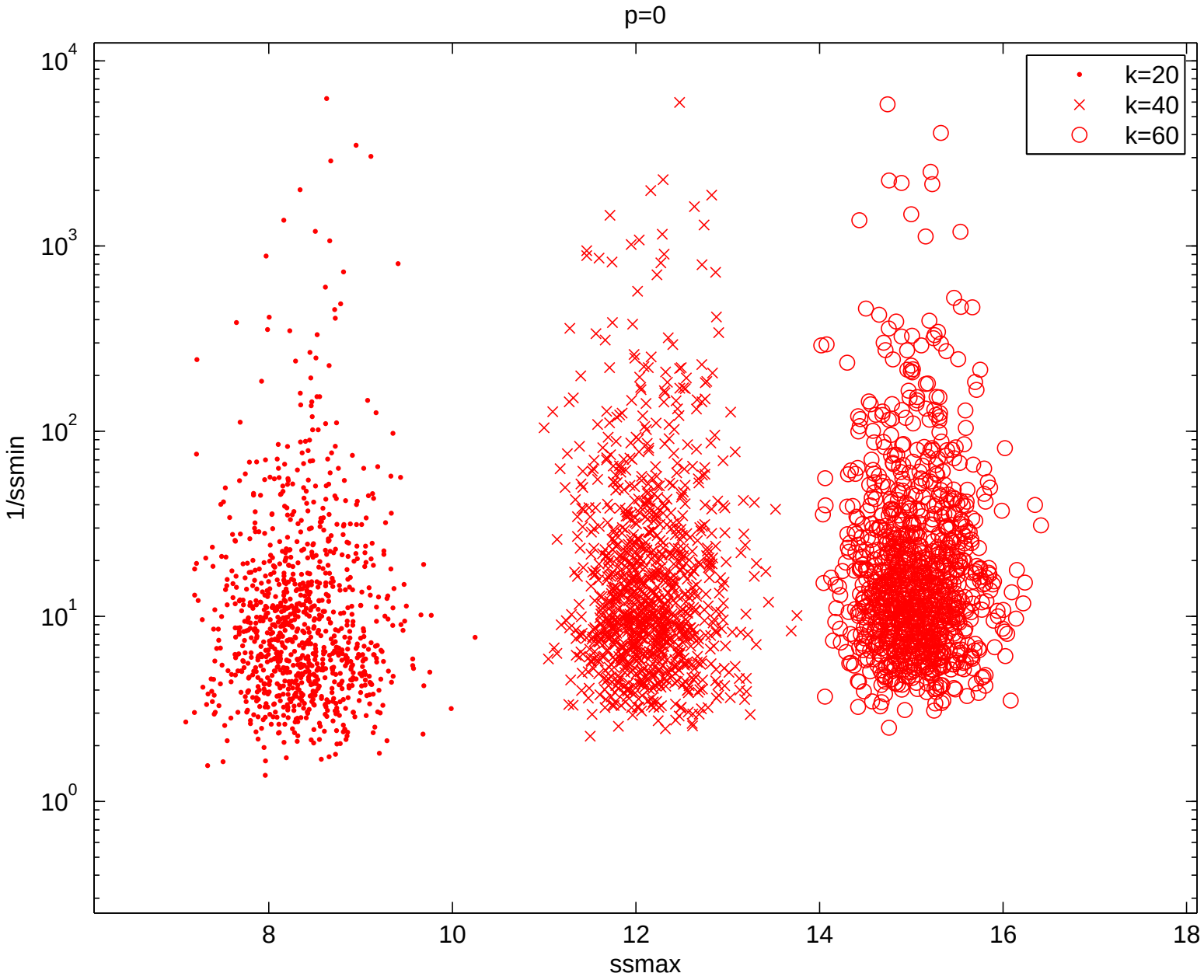
Assumption: $\mathbf{\Omega}$ is drawn from a normal Gaussian distribution.

Since the Gaussian distribution is rotationally invariant, the matrices $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ also have a Gaussian distribution. (As a consequence, the matrices **U** and **V** do not enter the analysis and one could simply assume that **A** is diagonal, $\mathbf{A} = \text{diag}(\sigma_1, \sigma_2, \ldots)$.)

What is the distribution of $\mathbf{\Omega}_1^\dagger$ when $\mathbf{\Omega}_1$ is a $k \times (k+p)$ Gaussian matrix?

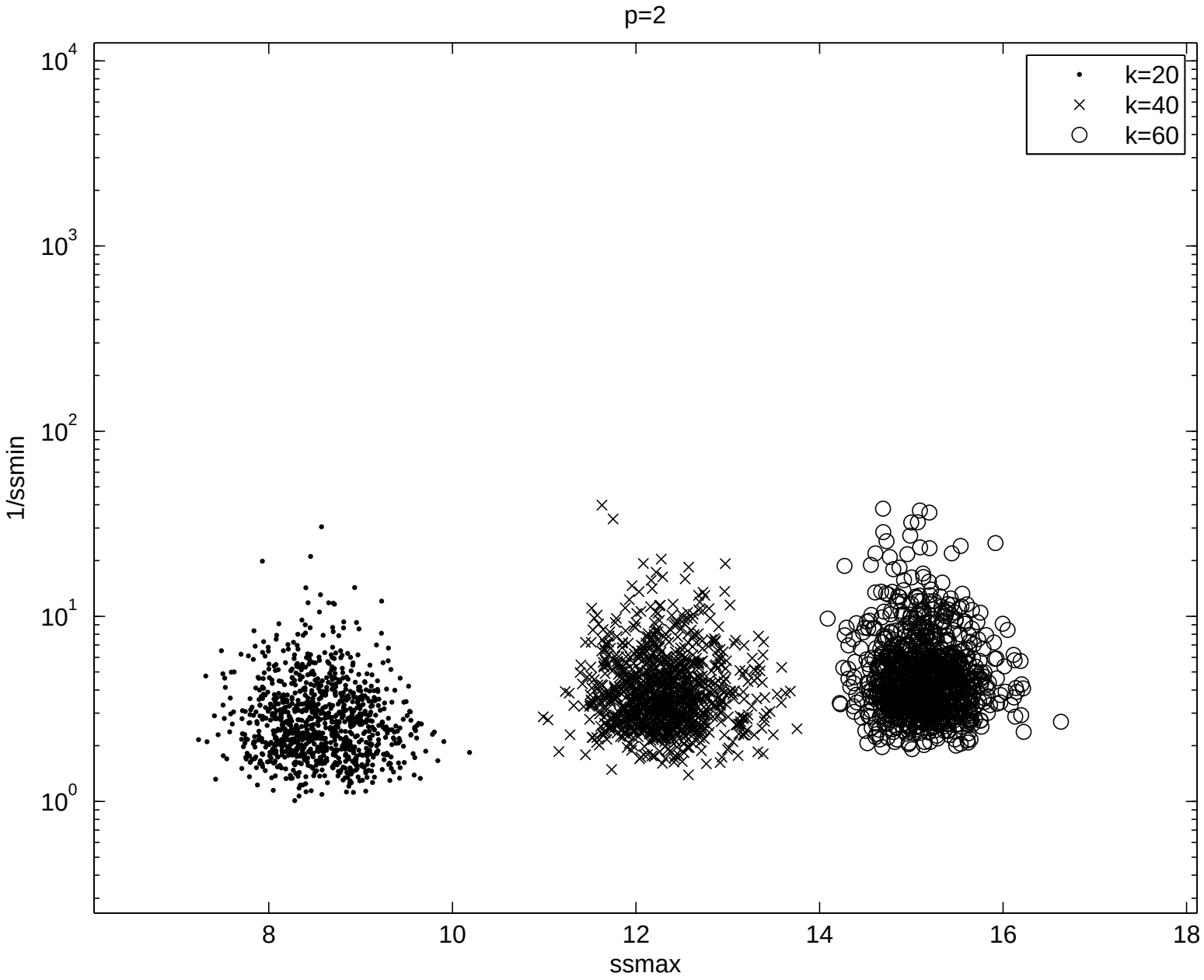
If $p = 0$, then $||\mathbf{\Omega}_1^\dagger||$ is typically large, and is very unstable.

Scatter plot showing distribution of $1/\sigma_{\min}$ for $k \times (k+p)$ Gaussian matrices. $p = 0$



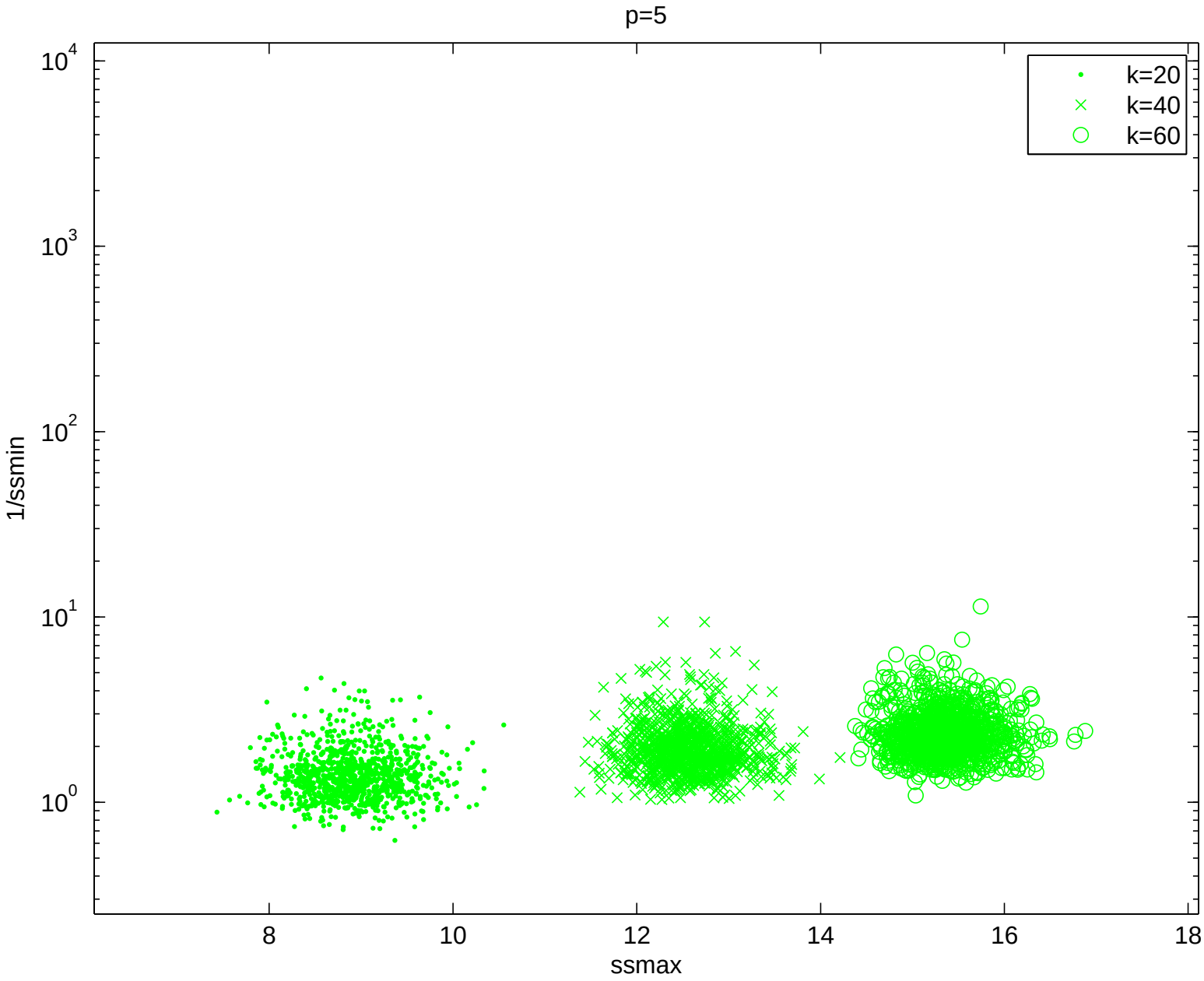
$1/\sigma_{\min}$ *is plotted against* $\sigma_{\max}$.

Scatter plot showing distribution of $1/\sigma_{\min}$ for $k \times (k+p)$ Gaussian matrices. $p = 2$



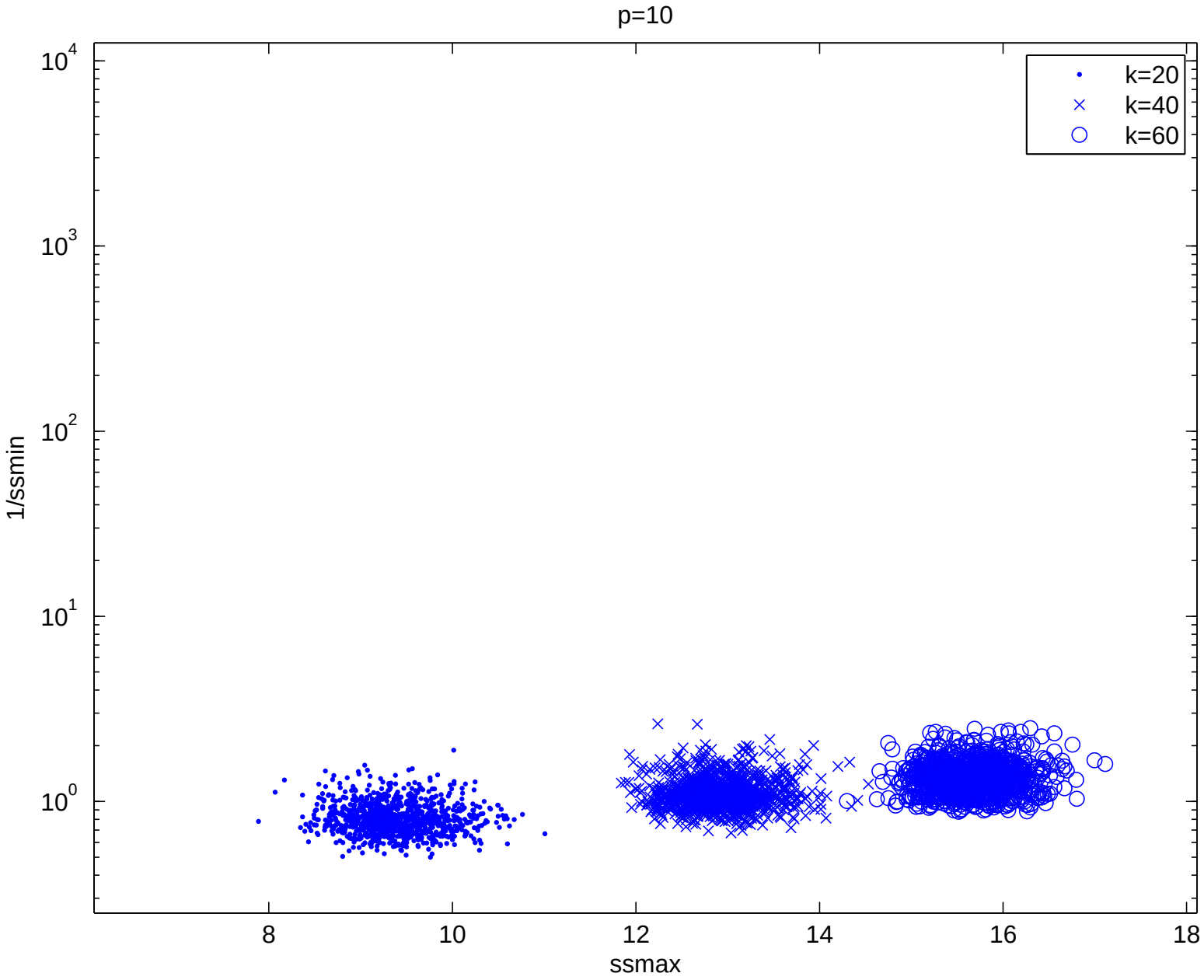
$1/\sigma_{\min}$ *is plotted against* $\sigma_{\max}$.

Scatter plot showing distribution of $1/\sigma_{\min}$ for $k \times (k+p)$ Gaussian matrices. $p = 5$



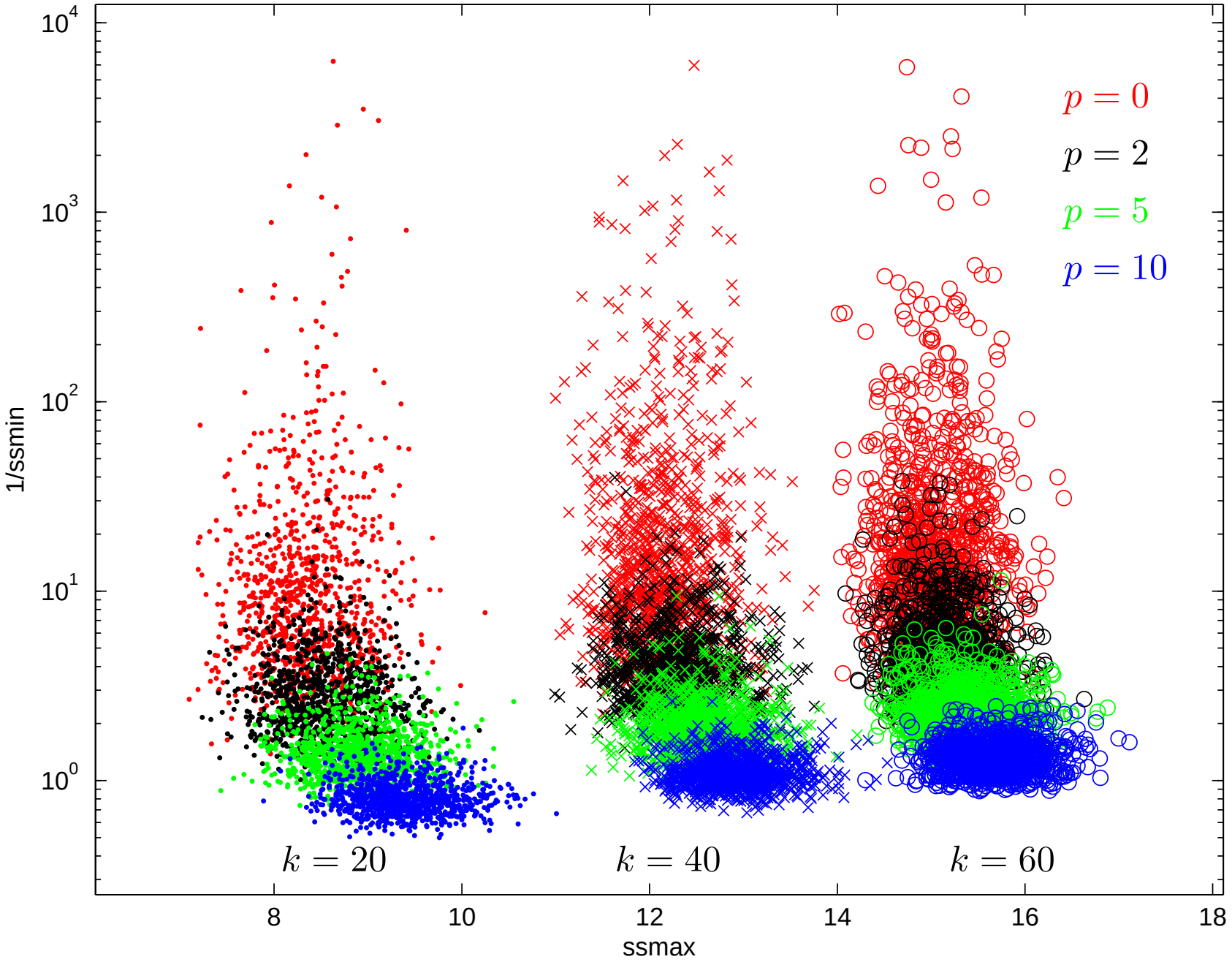
$1/\sigma_{\min}$ *is plotted against* $\sigma_{\max}$.

Scatter plot showing distribution of $1/\sigma_{\min}$ for $k \times (k+p)$ Gaussian matrices. $p = 10$



$1/\sigma_{\min}$ *is plotted against* $\sigma_{\max}$.

Scatter plot showing distribution of $k \times (k+p)$ Gaussian matrices.



$1/\sigma_{\min}$ *is plotted against* $\sigma_{\max}$.

Simplistic proof that a rectangular Gaussian matrix is well-conditioned:

Let **G** denote a $k \times \ell$ Gaussian matrix where $k < \ell$.
Let "g" denote a generic $\mathcal{N}(0,1)$ variable and "r_j^2" denote a generic χ_j^2 variable. Then

$$\mathbf{G} \sim \begin{bmatrix} g & g & g & g & g & g & \cdots \\ g & g & g & g & g & g & \cdots \\ g & g & g & g & g & g & \cdots \\ g & g & g & g & g & g & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \end{bmatrix} \sim \begin{bmatrix} r_\ell & 0 & 0 & 0 & 0 & 0 & \cdots \\ g & g & g & g & g & g & \cdots \\ g & g & g & g & g & g & \cdots \\ g & g & g & g & g & g & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \end{bmatrix}$$

$$\sim \begin{bmatrix} r_\ell & 0 & 0 & 0 & 0 & 0 & \cdots \\ r_{k-1} & g & g & g & g & g & \cdots \\ 0 & g & g & g & g & g & \cdots \\ 0 & g & g & g & g & g & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \end{bmatrix} \sim \begin{bmatrix} r_\ell & 0 & 0 & 0 & 0 & \cdots \\ r_{k-1} & r_{\ell-1} & 0 & 0 & 0 & \cdots \\ 0 & g & g & g & g & \cdots \\ 0 & g & g & g & g & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots \end{bmatrix}$$

$$\sim \begin{bmatrix} r_\ell & 0 & 0 & 0 & 0 & \cdots \\ r_{k-1} & r_{\ell-1} & 0 & 0 & 0 & \cdots \\ 0 & r_{k-2} & g & g & g & \cdots \\ 0 & 0 & g & g & g & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots \end{bmatrix} \sim \cdots \sim \begin{bmatrix} r_\ell & 0 & 0 & 0 & 0 & \cdots \\ r_{k-1} & r_{\ell-1} & 0 & 0 & 0 & \cdots \\ 0 & r_{k-2} & r_{\ell-2} & 0 & 0 & \cdots \\ 0 & 0 & r_{k-3} & r_{\ell-3} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots \end{bmatrix}$$

Gershgorin's circle theorem will now show that **G** is well-conditioned if, e.g., $\ell = 2k$.

More sophisticated methods are required to get to $\ell = k + 2$.

Some results on Gaussian matrices. *Adapted from HMT 2009/2011; Gordon (1985,1988) for Proposition 1; Chen & Dongarra (2005) for Propositions 2 and 4; Bogdanov (1998) for Proposition 3.*

Proposition 1: Let $\mathbf{G}$ be a Gaussian matrix. Then

$$\left(\mathbb{E}\left[||\mathbf{SGT}||_{\mathrm{F}}^2\right]\right)^{1/2} \leq ||\mathbf{S}||_{\mathrm{F}}\,||\mathbf{T}||_{\mathrm{F}} \tag{1}$$

$$\mathbb{E}\left[||\mathbf{SGT}||\right] \leq ||\mathbf{S}||\,||\mathbf{T}||_{\mathrm{F}} + ||\mathbf{S}||_{\mathrm{F}}\,||\mathbf{T}|| \tag{2}$$

Proposition 2: Let $\mathbf{G}$ be a Gaussian matrix of size $k \times k+p$ where $p \geq 2$. Then

$$\left(\mathbb{E}\left[||\mathbf{G}^{\dagger}||_{\mathrm{F}}^2\right]\right)^{1/2} \leq \sqrt{\frac{k}{p-1}} \tag{3}$$

$$\mathbb{E}\left[||\mathbf{G}^{\dagger}||\right] \leq \frac{e\sqrt{k+p}}{p}. \tag{4}$$

Proposition 3: Suppose h is Lipschitz $|h(\mathbf{X}) - h(\mathbf{Y})| \leq L||\mathbf{X}-\mathbf{Y}||_{\mathrm{F}}$ and $\mathbf{G}$ is Gaussian. Then

$$\mathbb{P}\left[h(\mathbf{G}) > \mathbb{E}[h(\mathbf{G})] + L\,u\right] \leq e^{-u^2/2}. \tag{5}$$

Proposition 4: Suppose $\mathbf{G}$ is Gaussian of size $k \times k+p$ with $p \geq 4$. Then for $t \geq 1$:

$$\mathbb{P}\left[||\mathbf{G}^{\dagger}||_{\mathrm{F}} \geq \sqrt{\frac{3k}{p+1}}t\right] \leq t^{-p} \tag{6}$$

$$\mathbb{P}\left[||\mathbf{G}^{\dagger}|| \geq \frac{e\sqrt{k+p}}{p+1}t\right] \leq t^{-(p+1)} \tag{7}$$

Recall: $||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}||^2 \leq ||\mathbf{\Sigma}_2||^2 + ||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger||^2$, where $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ are Gaussian and $\mathbf{\Omega}_1$ is $k \times k + p$.

Theorem: $\mathbb{E}\big[||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}||\big] \leq \left(1 + \sqrt{\frac{k}{p-1}}\right)\sigma_{k+1} + \frac{e\sqrt{k+p}}{p}\left(\sum_{j=k+1}^{\min(m,n)} \sigma_j^2\right)^{1/2}$.

Proof: First observe that

$$\mathbb{E}||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}|| = \mathbb{E}\big(||\mathbf{\Sigma}_2||^2 + ||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger||^2\big)^{1/2} \leq ||\mathbf{\Sigma}_2|| + \mathbb{E}||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger||.$$

Condition on $\mathbf{\Omega}_1$ and use Proposition 1:

$$\mathbb{E}||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger|| \leq \mathbb{E}\big[||\mathbf{\Sigma}_2||\,||\mathbf{\Omega}_1^\dagger||_{\mathrm{F}} + ||\mathbf{\Sigma}_2||_{\mathrm{F}}\,||\mathbf{\Omega}_1^\dagger||\big]$$

$$\leq \{\text{Hölder}\} \leq ||\mathbf{\Sigma}_2||\left(\mathbb{E}||\mathbf{\Omega}_1^\dagger||_{\mathrm{F}}^2\right)^{1/2} + ||\mathbf{\Sigma}_2||_{\mathrm{F}}\,\mathbb{E}||\mathbf{\Omega}_1^\dagger||.$$

Proposition 2 now provides bounds for $\mathbb{E}||\mathbf{\Omega}_1^\dagger||_{\mathrm{F}}^2$ and $\mathbb{E}||\mathbf{\Omega}_1^\dagger||$ and we get

$$\mathbb{E}||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger|| \leq \sqrt{\frac{k}{p-1}}||\mathbf{\Sigma}_2|| + \frac{e\sqrt{k+p}}{p}||\mathbf{\Sigma}_2||_{\mathrm{F}} = \sqrt{\frac{k}{p-1}}\sigma_{k+1} + \frac{e\sqrt{k+p}}{p}\left(\sum_{j>k}\sigma_j^2\right)^{1/2}.$$

Some results on Gaussian matrices. *Adapted from HMT2009/2011; Gordon (1985,1988) for Proposition 1; Chen & Dongarra (2005) for Propositions 2 and 4; Bogdanov (1998) for Proposition 3.*

Proposition 1: Let **G** be a Gaussian matrix. Then

$$\left(\mathbb{E}\left[||\mathbf{SGT}||_{\mathrm{F}}^2\right]\right)^{1/2} \leq ||\mathbf{S}||_{\mathrm{F}}\,||\mathbf{T}||_{\mathrm{F}} \tag{8}$$

$$\mathbb{E}\left[||\mathbf{SGT}||\right] \leq ||\mathbf{S}||\,||\mathbf{T}||_{\mathrm{F}} + ||\mathbf{S}||_{\mathrm{F}}\,||\mathbf{T}|| \tag{9}$$

Proposition 2: Let **G** be a Gaussian matrix of size $k \times k+p$ where $p \geq 2$. Then

$$\left(\mathbb{E}\left[||\mathbf{G}^{\dagger}||_{\mathrm{F}}^2\right]\right)^{1/2} \leq \sqrt{\frac{k}{p-1}} \tag{10}$$

$$\mathbb{E}\left[||\mathbf{G}^{\dagger}||\right] \leq \frac{e\sqrt{k+p}}{p}. \tag{11}$$

Proposition 3: Suppose h is Lipschitz $|h(\mathbf{X}) - h(\mathbf{Y})| \leq L||\mathbf{X}-\mathbf{Y}||_{\mathrm{F}}$ and **G** is Gaussian. Then

$$\mathbb{P}\left[h(\mathbf{G}) > \mathbb{E}[h(\mathbf{G})] + L\,u\right] \leq e^{-u^2/2}. \tag{12}$$

Proposition 4: Suppose **G** is Gaussian of size $k \times k+p$ with $p \geq 4$. Then for $t \geq 1$:

$$\mathbb{P}\left[||\mathbf{G}^{\dagger}||_{\mathrm{F}} \geq \sqrt{\frac{3k}{p+1}}t\right] \leq t^{-p} \tag{13}$$

$$\mathbb{P}\left[||\mathbf{G}^{\dagger}|| \geq \frac{e\sqrt{k+p}}{p+1}t\right] \leq t^{-(p+1)} \tag{14}$$

Recall: $||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}||^2 \leq ||\mathbf{\Sigma}_2||^2 + ||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger||^2$, where $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ are Gaussian and $\mathbf{\Omega}_1$ is $k \times k+p$.

Theorem: With probability at least $1 - 2\,t^{-p} - e^{-u^2/2}$ it holds that

$$||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}|| \leq \left(1 + t\sqrt{\frac{3k}{p+1}} + u\,t\,\frac{e\,\sqrt{k+p}}{p+1}\right)\,\sigma_{k+1} + \frac{t\,e\,\sqrt{k+p}}{p+1}\left(\sum_{j>k}\sigma_j^2\right)^{1/2}.$$

Proof: Set $E_t = \left\{||\mathbf{\Omega}_1|| \leq \frac{e\sqrt{k+p}}{p+1}t \quad \text{and} \quad ||\mathbf{\Omega}_1^\dagger||_{\mathrm{F}} \leq \sqrt{\frac{3k}{p+1}}\,t\right\}$. By Proposition 4: $\mathbb{P}(E_t^{\mathrm{c}}) \leq 2\,t^{-p}$.

Set $h(\mathbf{X}) = ||\mathbf{\Sigma}_2\mathbf{X}\mathbf{\Omega}_1^\dagger||$. A direct calculation shows

$$|h(\mathbf{X}) - h(\mathbf{Y})| \leq ||\mathbf{\Sigma}_2||\,||\mathbf{\Omega}_1^\dagger||\,||\mathbf{X} - \mathbf{y}||_{\mathrm{F}}.$$

Hold Ω_1 fixed and take the expectation on Ω_2. Then Proposition 1 applies and so

$$\mathbb{E}\big[h(\mathbf{\Omega}_2) \;\big|\; \mathbf{\Omega}_1\big] \leq ||\mathbf{\Sigma}_2||\,||\mathbf{\Omega}_1^\dagger||_{\mathrm{F}} + ||\mathbf{\Sigma}_2||_{\mathrm{F}}\,||\mathbf{\Omega}_1^\dagger||.$$

Now use Proposition 3 (concentration of measure)

$$\mathbb{P}\Big[\underbrace{||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger||}_{=h(\mathbf{\Omega}_2)} > \underbrace{||\mathbf{\Sigma}_2||\,||\mathbf{\Omega}_1^\dagger||_{\mathrm{F}} + ||\mathbf{\Sigma}_2||_{\mathrm{F}}\,||\mathbf{\Omega}_1^\dagger||}_{=\mathbb{E}[h(\mathbf{\Omega}_2)]} + \underbrace{||\mathbf{\Sigma}_2||\,||\mathbf{\Omega}_1^\dagger||}_{=L}\,u \;\Big|\; E_t\Big] < e^{-u^2/2}.$$

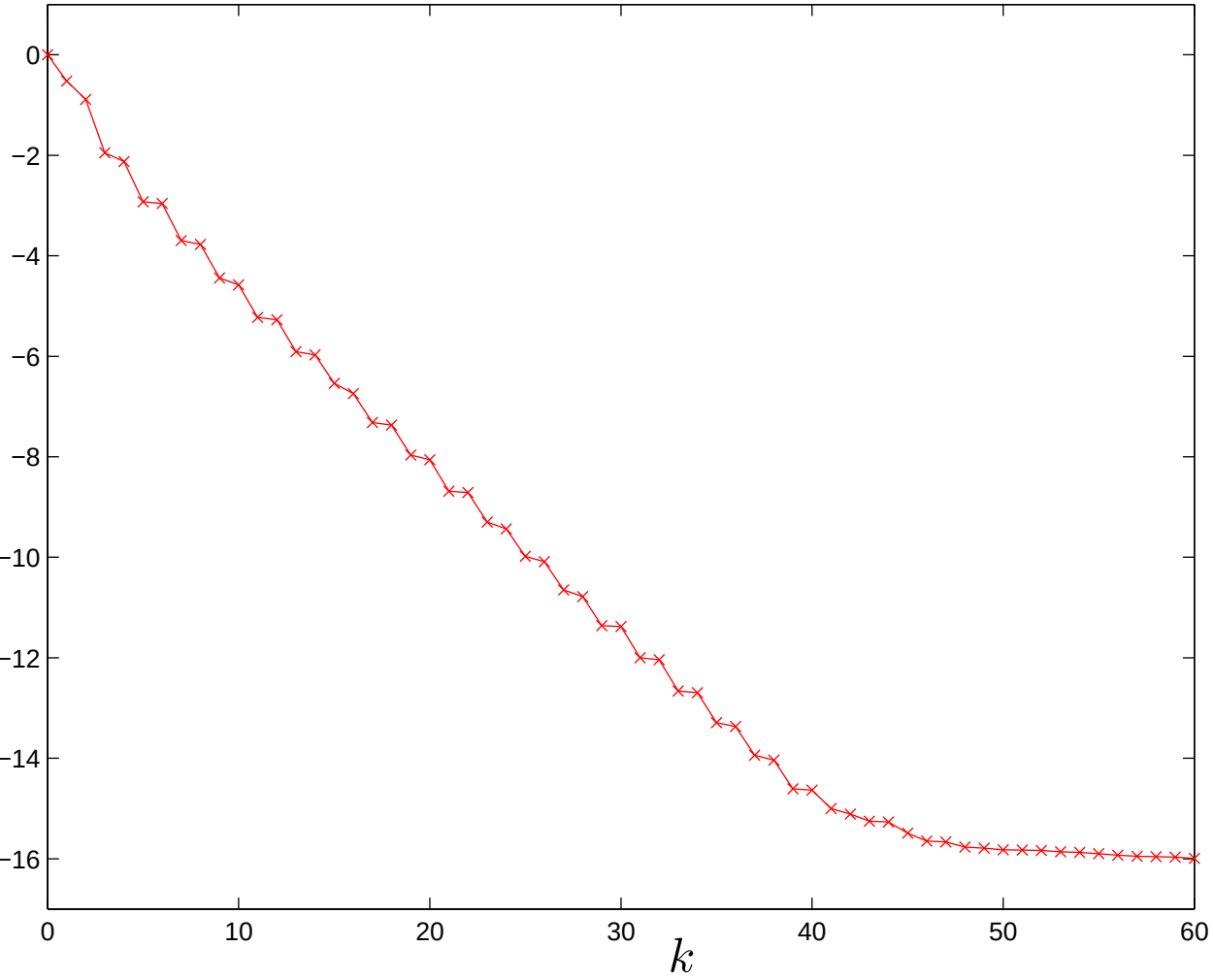
When E_t holds true, we have bounds on the "badness" of $\Omega_1^\dagger$:

$$\mathbb{P}\Big[||\mathbf{\Sigma}_2\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger|| > ||\mathbf{\Sigma}_2||\sqrt{\frac{3k}{p+1}}t + ||\mathbf{\Sigma}_2||_{\mathrm{F}}\,\frac{e\sqrt{k+p}}{p+1}t + ||\mathbf{\Sigma}_2||\,\frac{e\sqrt{k+p}}{p+1}\,ut \;\Big|\; E_t\Big] < e^{-u^2/2}.$$

The theorem is obtained by using $\mathbb{P}(E_t^{\mathrm{c}}) \leq 2\,t^{-p}$ to remove the conditioning of E_t.

Example 1:

We consider a $1\,000 \times 1\,000$ matrix **A** whose singular values are shown below:

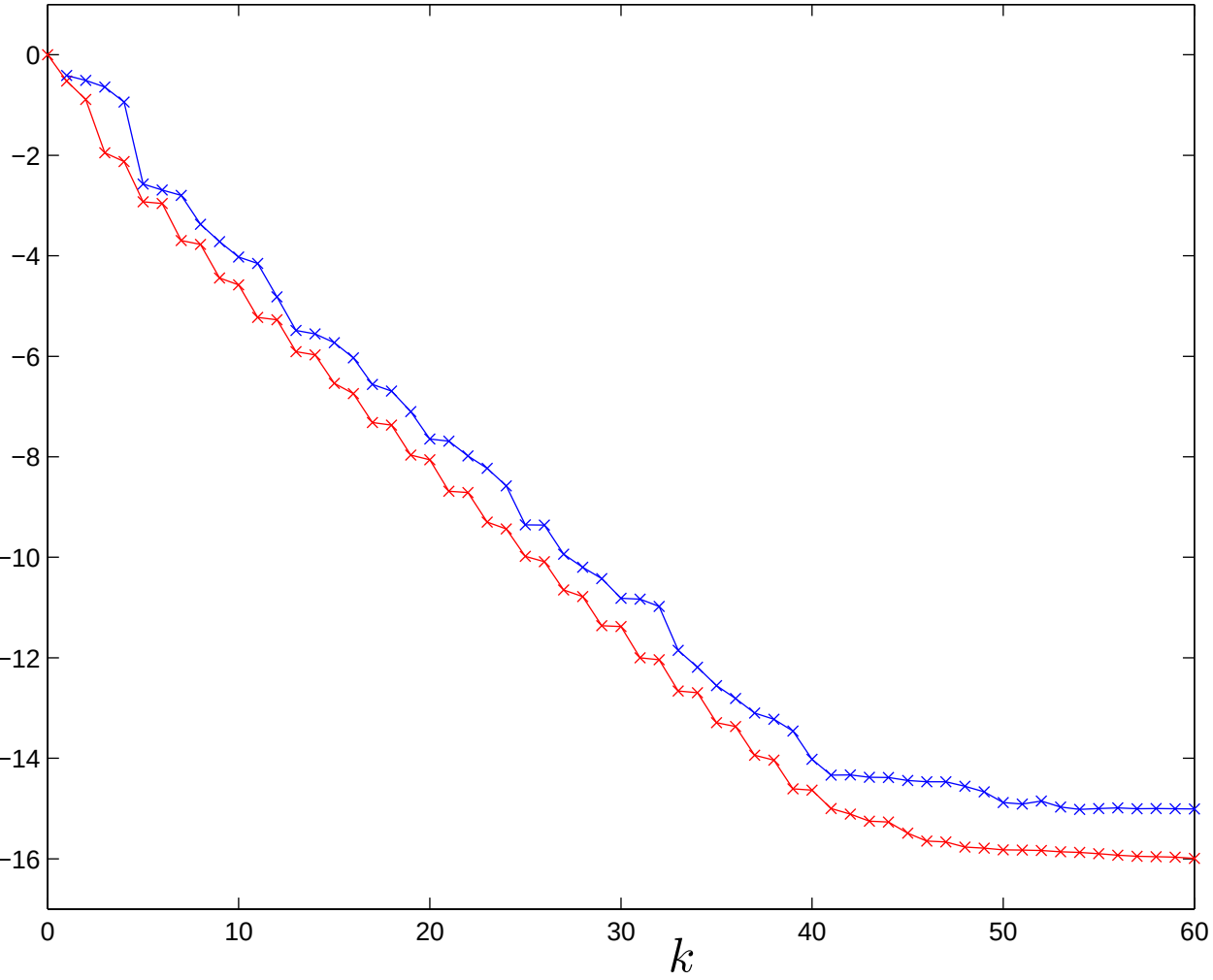


The red line indicates the singular values σ_{k+1} of **A**. These indicate the theoretically minimal approximation error.

A is a discrete approximation of a certain compact integral operator. Curiously, the nature of **A** is in a strong sense irrelevant: the error distribution depends only on $(\sigma_j)_{j=1}^{\min(m,n)}$.

Example 1:

We consider a $1\,000 \times 1\,000$ matrix **A** whose singular values are shown below:



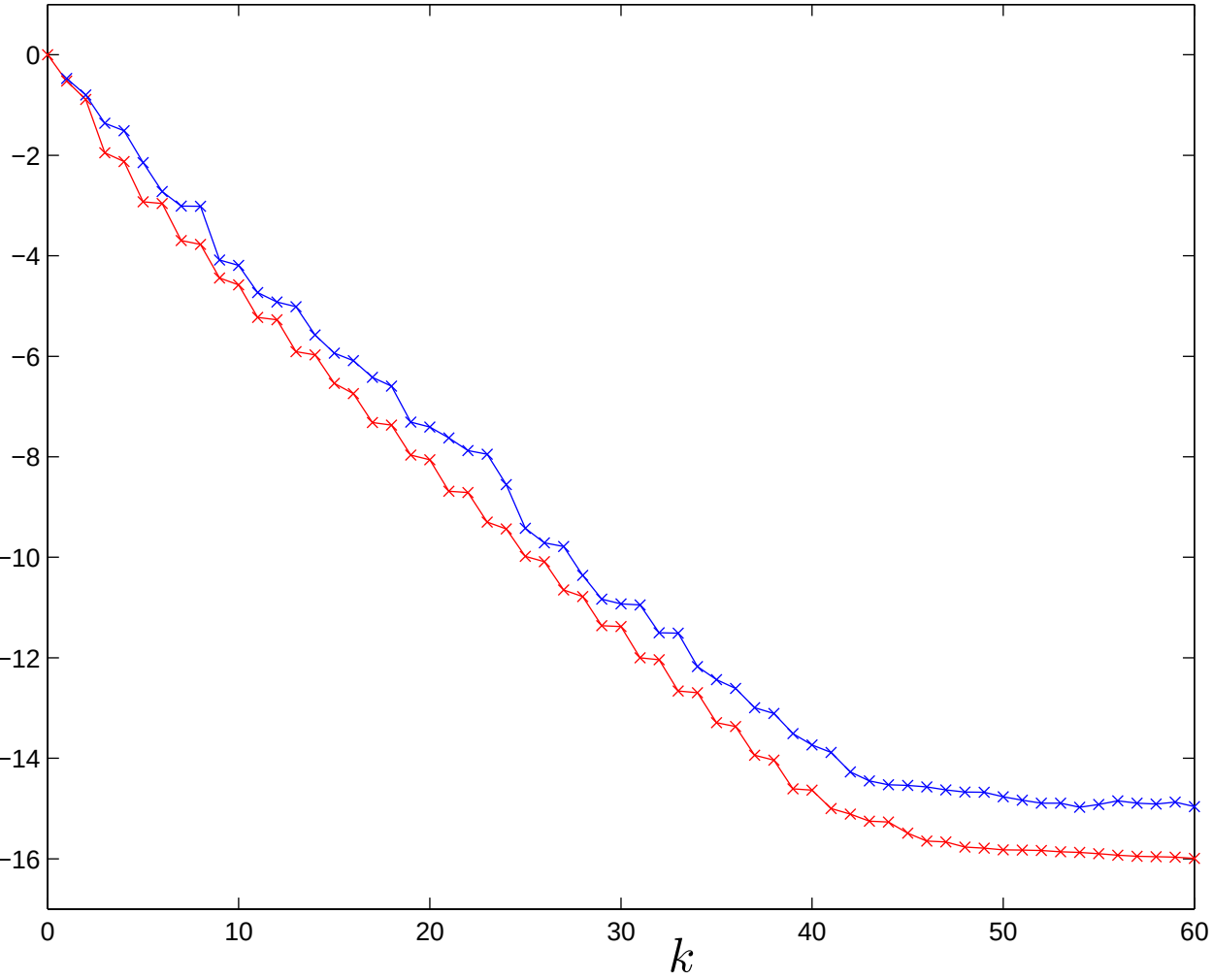
The red line indicates the singular values σ_{k+1} of **A**. These indicate the theoretically minimal approximation error.

The blue line indicates the actual errors e_k incurred by one instantiation of the proposed method.

A is a discrete approximation of a certain compact integral operator. Curiously, the nature of **A** is in a strong sense irrelevant: the error distribution depends only on $(\sigma_j)_{j=1}^{\min(m,n)}$.

Example 1:

We consider a $1\,000 \times 1\,000$ matrix **A** whose singular values are shown below:



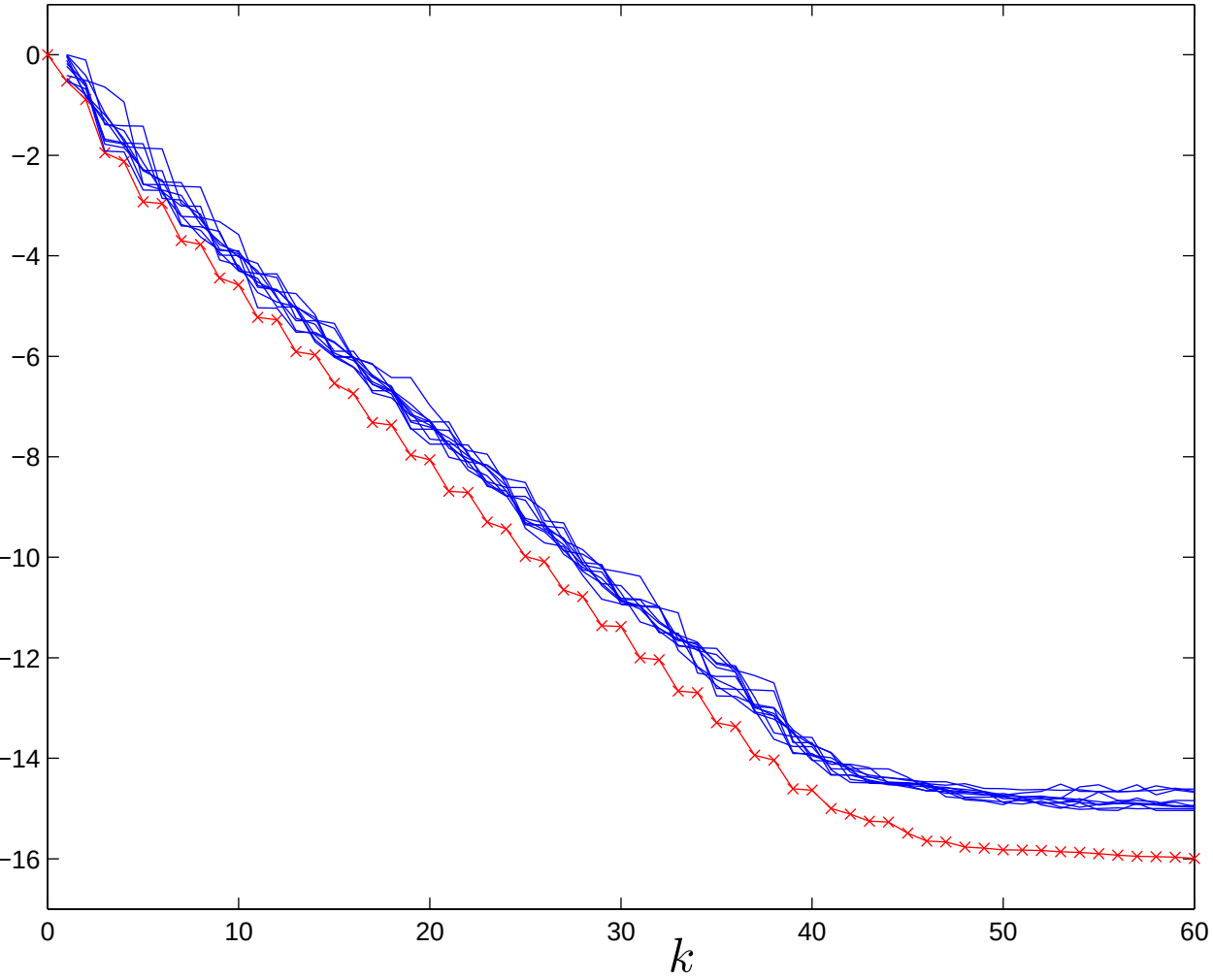
The red line indicates the singular values σ_{k+1} of **A**. These indicate the theoretically minimal approximation error.

The blue line indicates the actual errors e_k incurred by a different instantiation of the proposed method.

A is a discrete approximation of a certain compact integral operator. Curiously, the nature of **A** is in a strong sense irrelevant: the error distribution depends only on $(\sigma_j)_{j=1}^{\min(m,n)}$.

Example 1:

We consider a $1\,000 \times 1\,000$ matrix **A** whose singular values are shown below:



The red line indicates the singular values σ_{k+1} of **A**. These indicate the theoretically minimal approximation error.

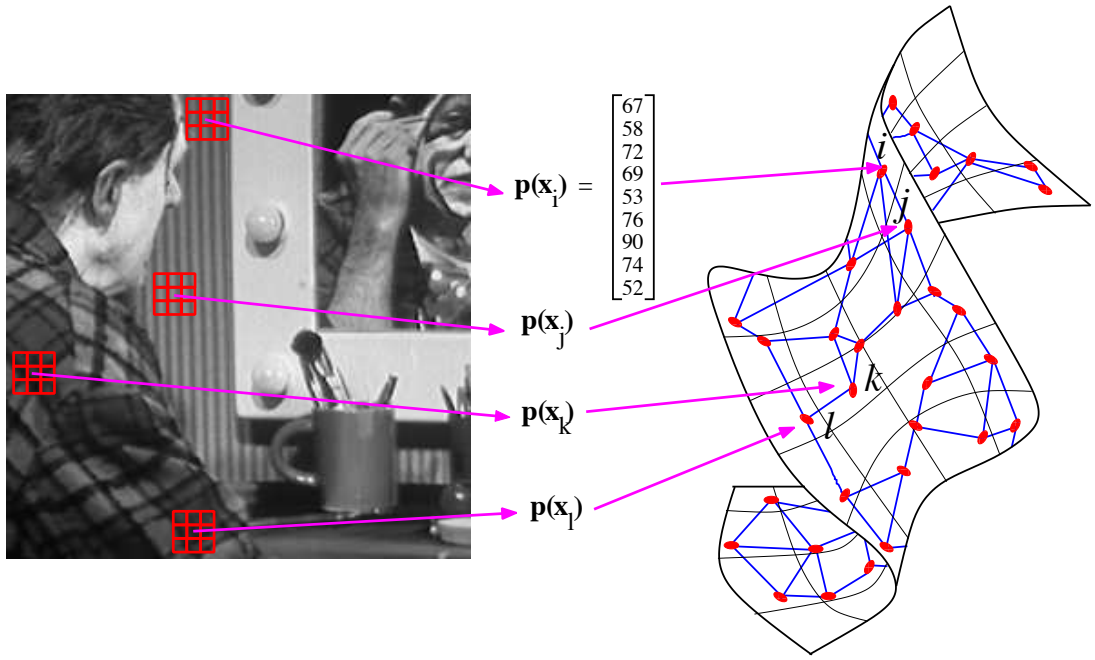
The blue lines indicate the actual errors e_k incurred by 10 different instantiations of the proposed method.

A is a discrete approximation of a certain compact integral operator. Curiously, the nature of **A** is in a strong sense irrelevant: the error distribution depends only on $(\sigma_j)_{j=1}^{\min(m,n)}$.

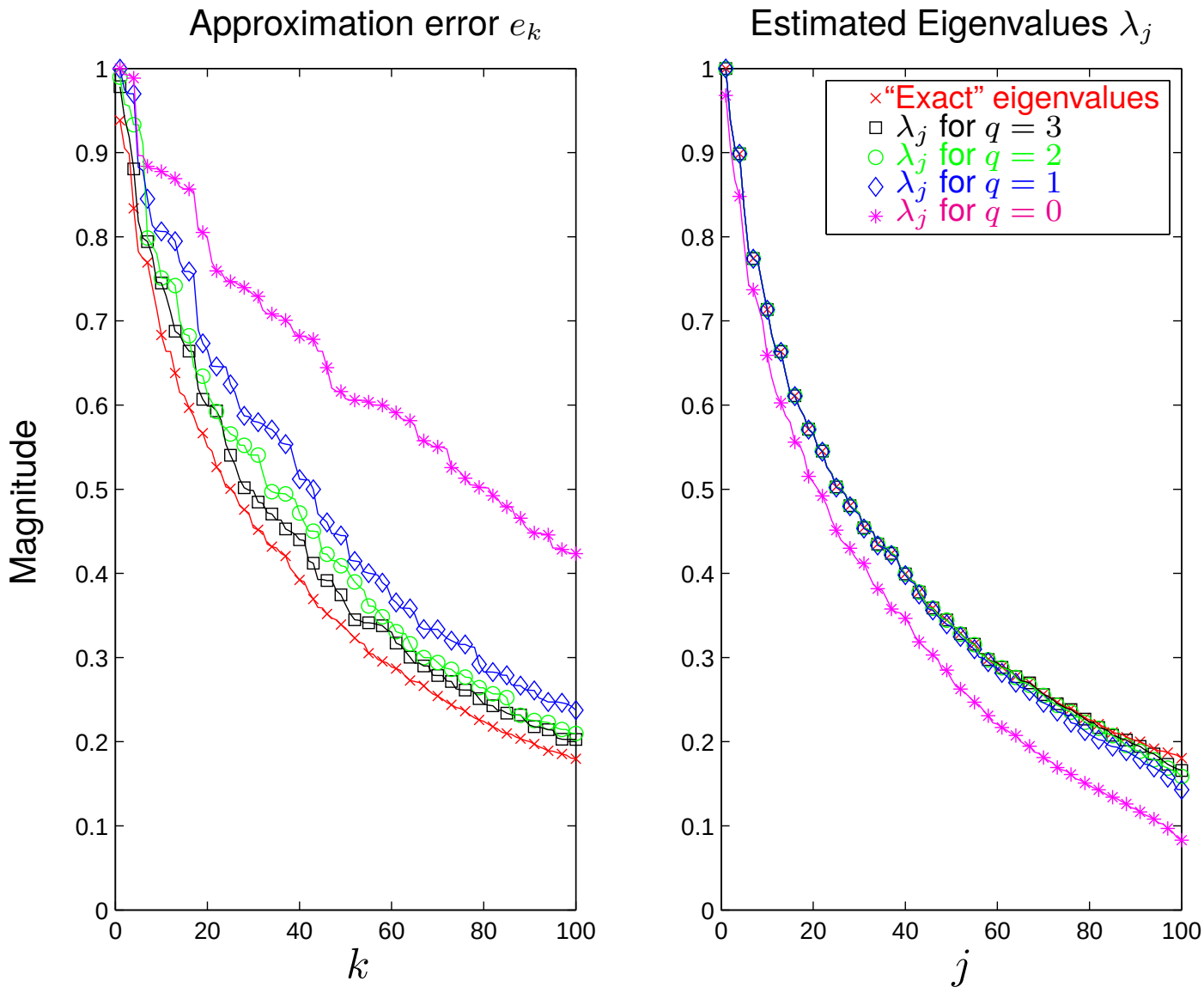
Example 2:

The matrix **A** being analyzed is a 9025×9025 matrix arising in a diffusion geometry approach to image processing.

To be precise, **A** is a graph Laplacian on the manifold of 3×3 patches.



Joint work with François Meyer of the University of Colorado at Boulder.



The pink lines illustrates the performance of the basic random sampling scheme.

The errors are huge, and the estimated eigenvalues are much too small.

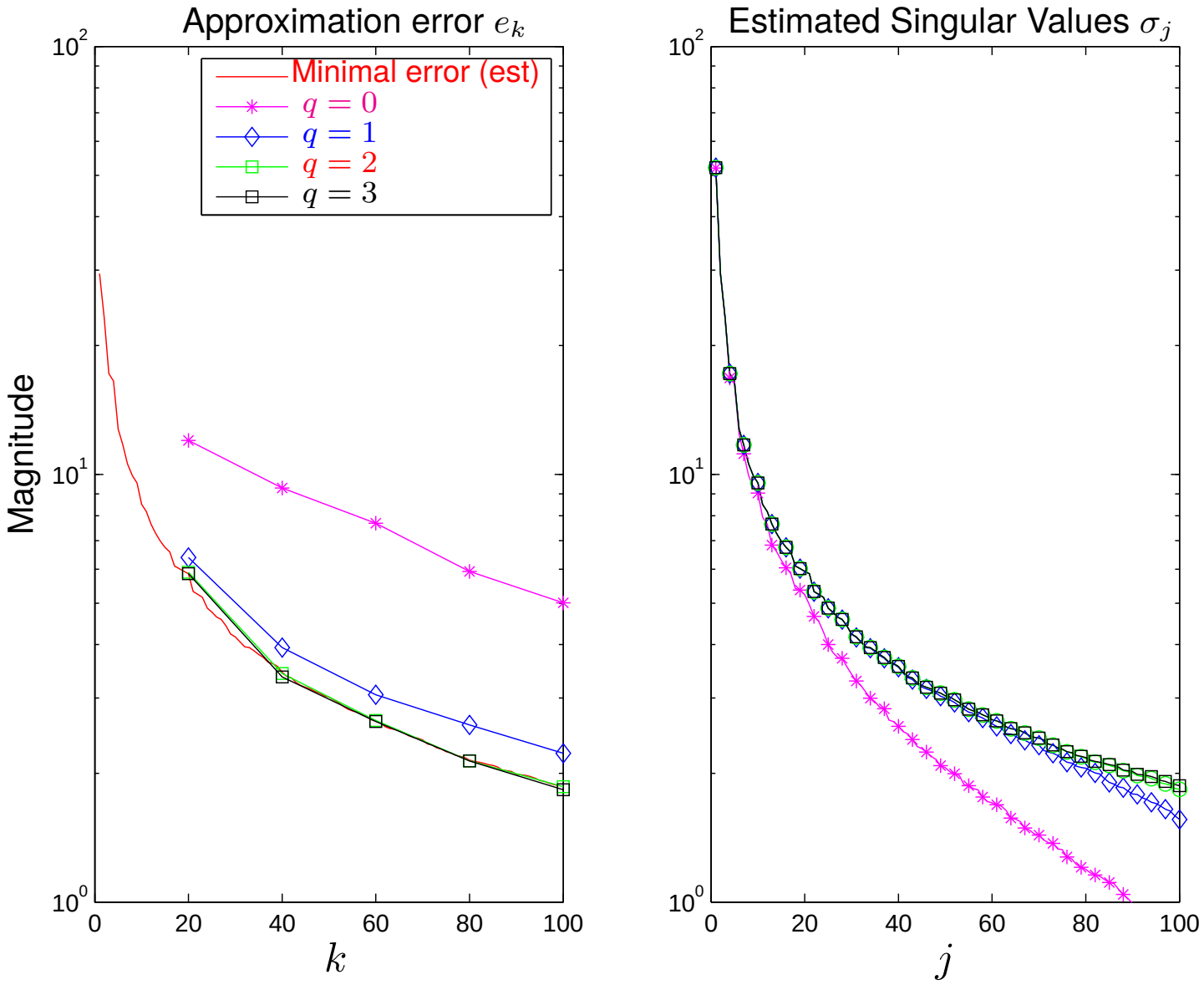
Example 3: “Eigenfaces”

We next process a data base containing $m = 7\,254$ pictures of faces

Each image consists of $n = 384 \times 256 = 98\,304$ gray scale pixels.

We center and scale the pixels in each image, and let the resulting values form a column of a $98\,304 \times 7\,254$ data matrix **A**.

The left singular vectors of **A** are the so called *eigenfaces* of the data base.



The pink lines illustrates the performance of the basic random sampling scheme. Again, the errors are huge, and the estimated eigenvalues are much too small.

Power method for improving accuracy:

The error depends on how quickly the singular values decay. Recall that

$$\mathbb{E}||\mathbf{A} - \mathbf{Q}\mathbf{Q}^*\mathbf{A}|| \leq \left(1 + \sqrt{\frac{k}{p-1}}\right)\sigma_{k+1} + \frac{e\sqrt{k+p}}{p}\left(\sum_{j=k+1}^{\min(m,n)} \sigma_j^2\right)^{1/2}.$$

The faster the singular values decay — the stronger the relative weight of the dominant modes in the samples.

Idea: The matrix $(\mathbf{A}\,\mathbf{A}^*)^q\,\mathbf{A}$ has the same left singular vectors as $\mathbf{A}$, and singular values

$$\sigma_j((\mathbf{A}\,\mathbf{A}^*)^q\,\mathbf{A}) = (\sigma_j(\mathbf{A}))^{2\,q+1}.$$

Much faster decay — so let us use the sample matrix

$$\mathbf{Y} = (\mathbf{A}\,\mathbf{A}^*)^q\,\mathbf{A}\,\mathbf{\Omega}$$

instead of

$$\mathbf{Y} = \mathbf{A}\,\mathbf{\Omega}.$$

References: Paper by Rokhlin, Szlam, Tygert (2008). Suggestions by Ming Gu. Also similar to “block power method,” “block Lanczos,” “subspace iteration.”

Input: An $m \times n$ matrix **A**, a target rank ℓ, and a small integer q.	
Output: Rank-ℓ factors **U**, **Σ**, and **V** in an approximate SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.	
(1) Draw an $n \times \ell$ random matrix **Ω**.	(4) Form the small matrix $\mathbf{B} = \mathbf{Q}^* \mathbf{A}$.
(2) Form the $n \times \ell$ sample matrix $\mathbf{Y} = (\mathbf{A}\,\mathbf{A}^*)^q\,\mathbf{A\Omega}$.	(5) Factor the small matrix $\mathbf{B} = \hat{\mathbf{U}}\mathbf{\Sigma V}^*$.
(3) Compute an ON matrix **Q** s.t. $\mathbf{Y} = \mathbf{QQ}^*\mathbf{Y}$.	(6) Form $\mathbf{U} = \mathbf{Q}\hat{\mathbf{U}}$.

- Detailed (and, we believe, close to sharp) error bounds have been proven.
  For instance, the expectation of the error satisfies:

$$\mathbb{E}\left[\,||\mathbf{A} - \mathbf{A}_{k+p}^{\text{computed}}||\,\right] \leq \left(1 + 4\sqrt{\frac{2\,\min(m,n)}{k-1}}\right)^{1/(2q+1)} \sigma_{k+1}(\mathbf{A}). \tag{15}$$

  Reference: Halko, Martinsson, Tropp (2011).

- The improved accuracy from the modified scheme comes at a cost;
  $2q+1$ passes over the matrix are required instead of 1.
  However, q can often be chosen quite small in practice, $q = 2$ or $q = 3$, say.

- The bound (15) assumes exact arithmetic.
  To handle round-off errors, variations of subspace iterations can be used.
  These are entirely numerically stable and achieve the same error bound.

A numerically stable version of the "power method":

Input: An $m \times n$ matrix $\mathbf{A}$, a target rank ℓ, and a small integer q.
Output: Rank-ℓ factors $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{V}$ in an approximate SVD $\mathbf{A} \approx \mathbf{U\Sigma V}^*$.

Draw an $n \times \ell$ Gaussian random matrix $\mathbf{\Omega}$.
Set $\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega})$
for $i = 1, 2, \ldots, q$
    $\mathbf{W} = \texttt{orth}(\mathbf{A}^*\mathbf{Q})$
    $\mathbf{Q} = \texttt{orth}(\mathbf{AW})$
end for
$\mathbf{B} = \mathbf{Q}^*\mathbf{A}$
$[\hat{\mathbf{U}}, \mathbf{\Sigma}, \mathbf{V}] = \texttt{svd}(\mathbf{B})$
$\mathbf{U} = \mathbf{Q}\hat{\mathbf{U}}$.

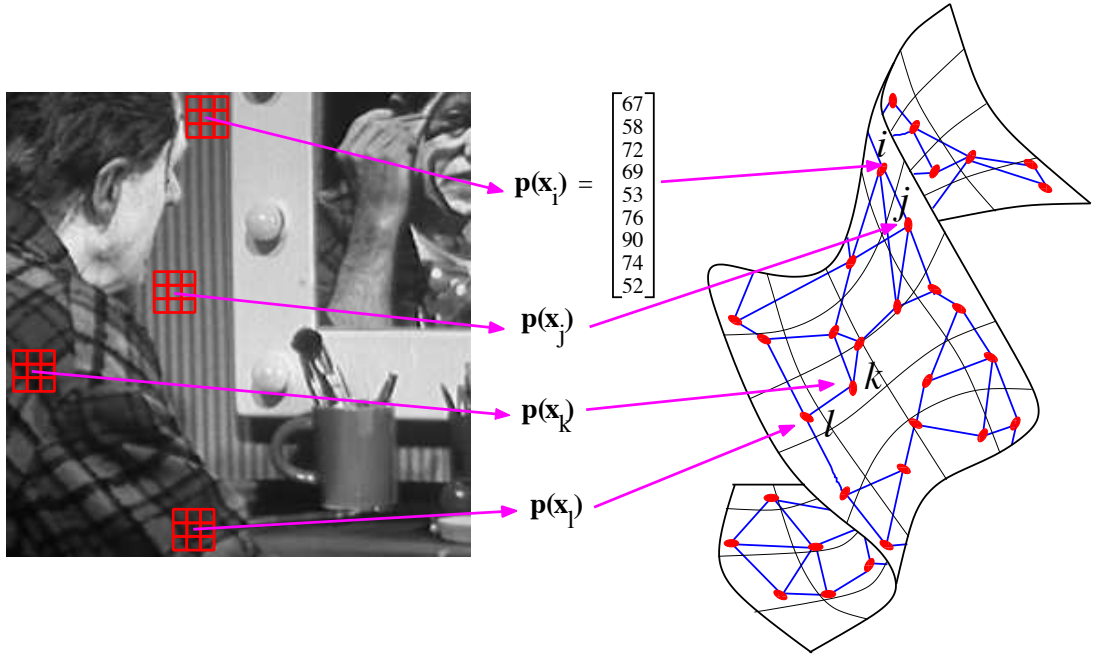
Note: Algebraically, the method with orthogonalizations is identical to the "original" method where $\mathbf{Q} = \texttt{orth}((\mathbf{AA}^*)^q\mathbf{A\Omega})$.

Note: This is a classic subspace iteration.
The novelty is the error analysis, and the finding that using a very small q is often fine.
(In fact, our analysis allows q to be zero...)

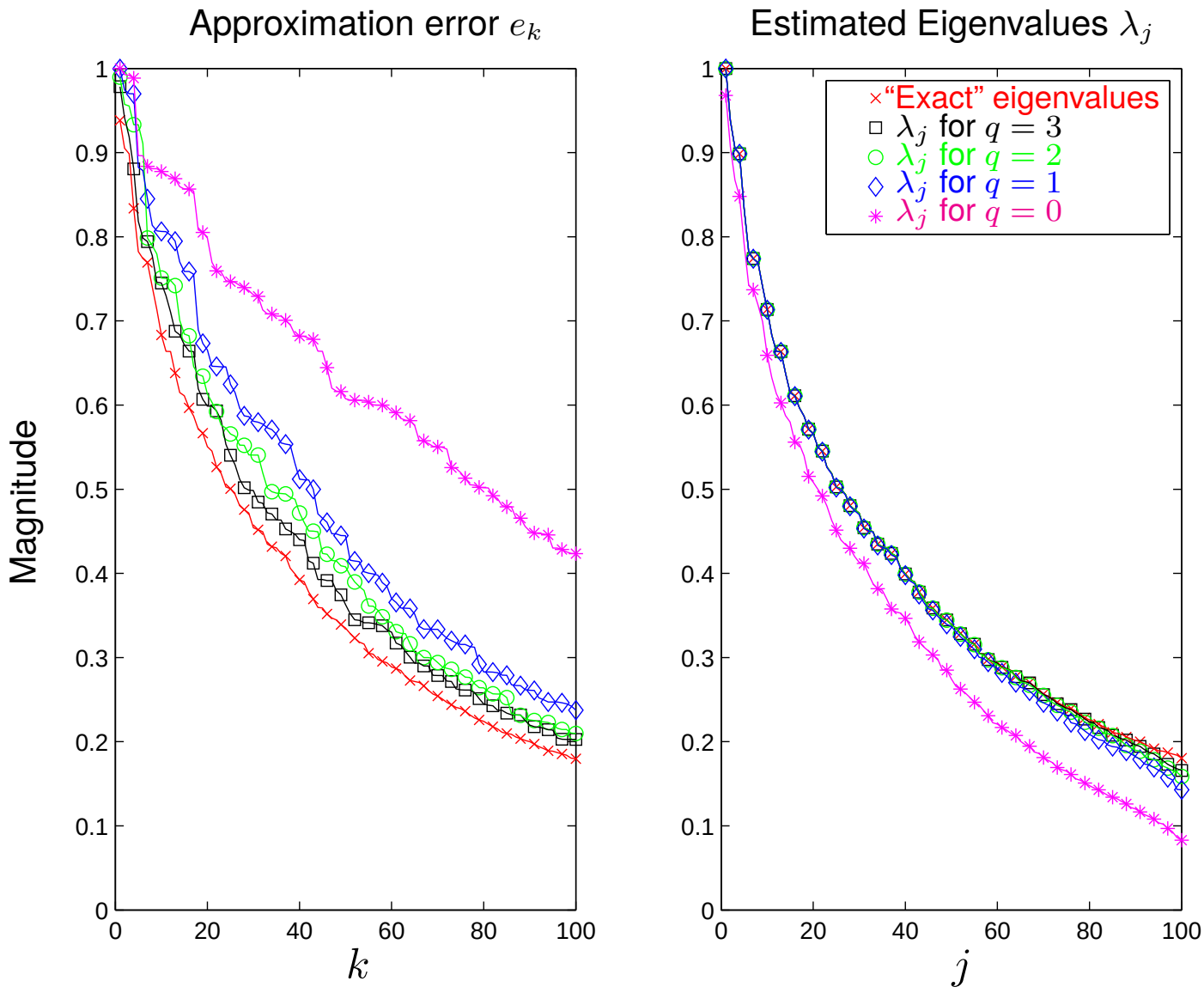
Example 2 (revisited):

The matrix **A** being analyzed is a 9025×9025 matrix arising in a diffusion geometry approach to image processing.

To be precise, **A** is a graph Laplacian on the manifold of 3×3 patches.



Joint work with François Meyer of the University of Colorado at Boulder.



The pink lines illustrates the performance of the basic random sampling scheme.

The errors are huge, and the estimated eigenvalues are much too small.

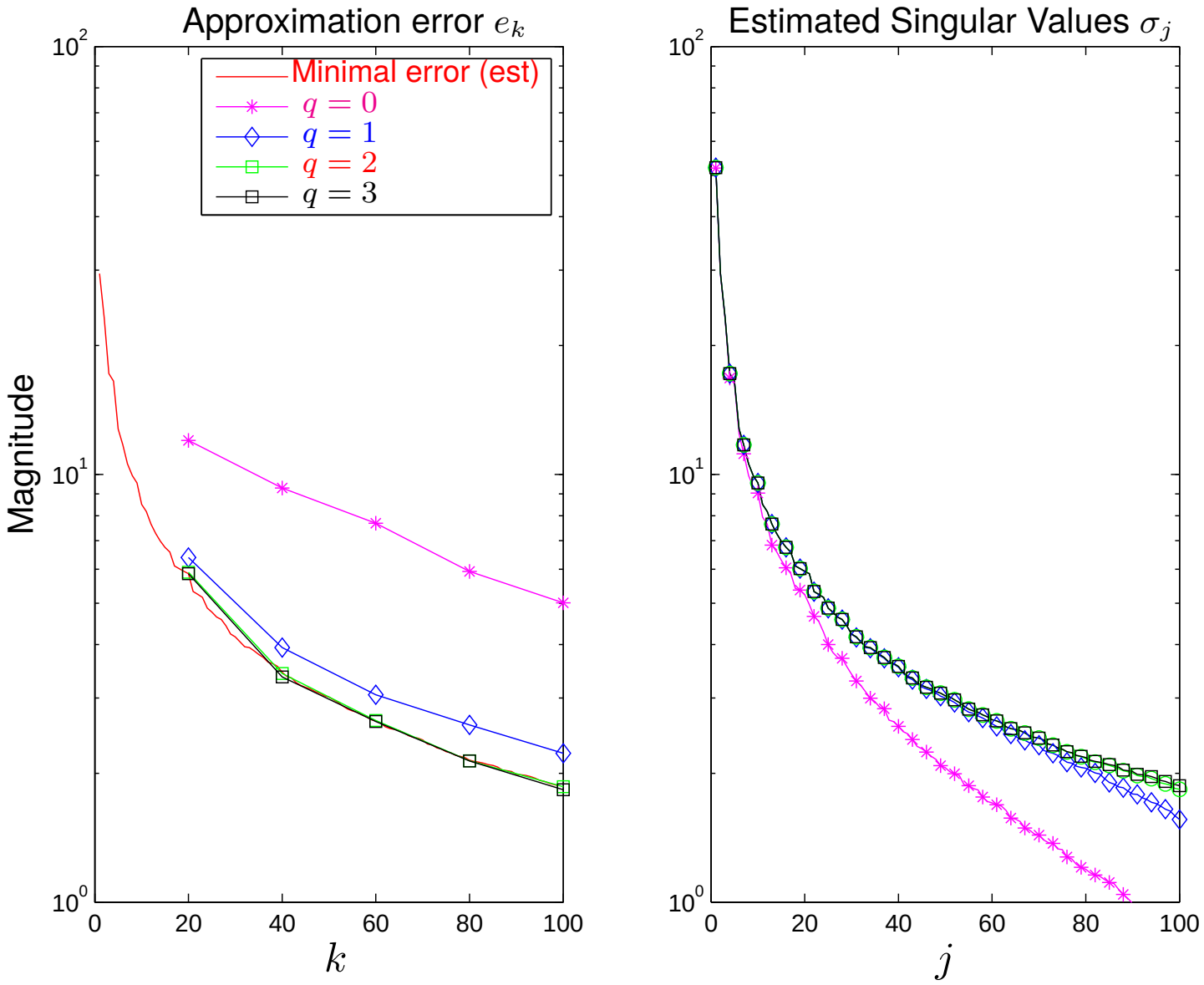
Example 3 (revisited): "Eigenfaces"

We next process a data base containing $m = 7\,254$ pictures of faces

Each image consists of $n = 384 \times 256 = 98\,304$ gray scale pixels.

We center and scale the pixels in each image, and let the resulting values form a column of a $98\,304 \times 7\,254$ data matrix **A**.

The left singular vectors of **A** are the so called *eigenfaces* of the data base.



The pink lines illustrates the performance of the basic random sampling scheme. Again, the errors are huge, and the estimated eigenvalues are much too small.

Current work

1. ***Package implementations:***
   Many of the randomized methods described are implemented in the (free) package "ID":
   `http://cims.nyu.edu/~tygert/software.html`
   In progress: Fuller multi-core support, GPU support, faster structured transforms, etc.

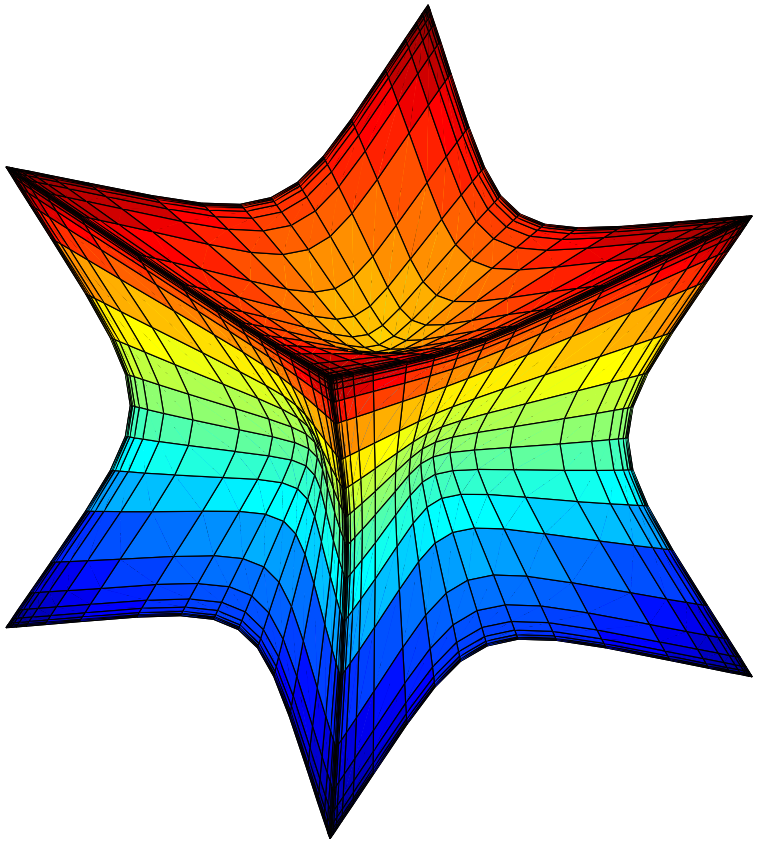
2. ***Integrate with applications:***
   Data mining, eigen-value problems on graphs, geo-tomography, etc.
   Workshop at ICERM at Brown: `http://icerm.brown.edu/sp-s14`

3. ***Structured matrix computations:***
   Many matrices that arise in applications have *off-diagonal blocks* that are of low rank:
   - Matrices approximating integral equations associated with elliptic PDEs.
   - Inverses of (sparse) matrices arising upon FEM discretization of elliptic PDEs.
   - Scattering matrices in acoustic and electro-magnetic scattering.
   - Buzzwords: $\mathcal{H}$-matrices, HSS-matrices, quasi-separable matrices, ...

   Using randomized algorithms, we have developed $O(N)$-complexity methods for performing algebraic operations on dense matrices of this type. This leads to:

   Accelerated direct solvers for elliptic PDEs.
   $O(N)$ complexity in many situations.

Numerical example — BIE on "edgy" surface (2013, with J. Bremer and A. Gillman)

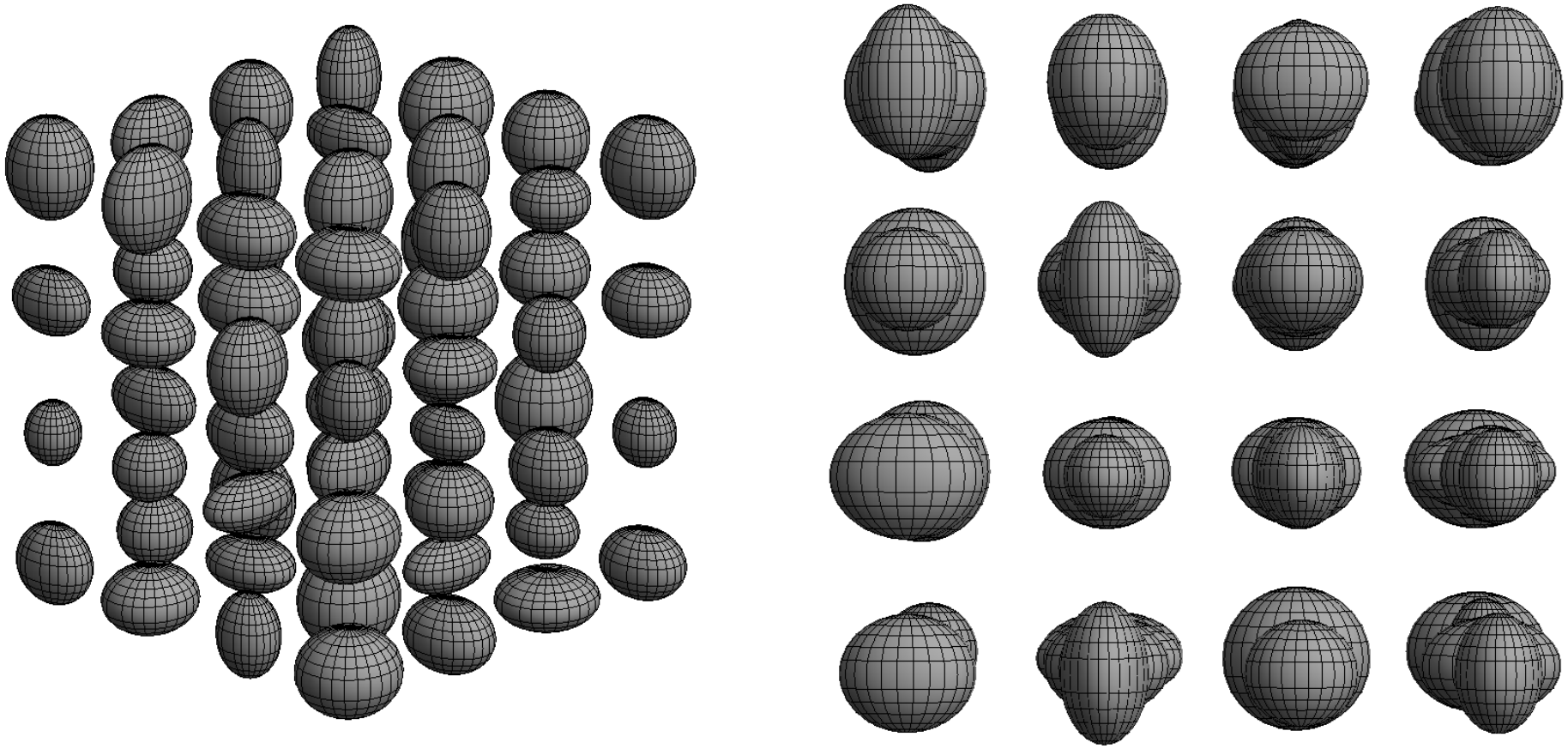


N_{tris}	N	E	T	$N_{\text{out}} \times N_{\text{in}}$
192	$21\,504$	2.60×10^{-08}	$6.11 \times 10^{+02}$	617×712
432	$48\,384$	2.13×10^{-09}	$1.65 \times 10^{+03}$	620×694
768	$86\,016$	3.13×10^{-10}	$3.58 \times 10^{+03}$	612×685

Results from a Helmholtz problem (acoustic scattering) on the domain exterior to the "edgy" cube.
The domain is about 3.5 wave-lengths in diameter.
Mesh refinement near edges & corners. The direct solver eliminates "extra" DOFs.

Numerical example — BIE on surfaces in 3D (2013, with J. Bremer and A. Gillman)

Consider sound-soft scattering from a multi-body scatterer of size 4 wave-lengths:



The ellipsoids are not rotationally symmetric.

The global scattering matrix is computed using the HBS direct solver described.

Numerical example — BIE on surfaces in 3D (2013, with J. Bremer and A. Gillman)

The local truncation error is set to 10^{-3}.

Grid dimensions	N	$N_{\text{out}} \times N_{\text{in}}$	T	E	Ratio	Predicted
$2 \times 2 \times 2$	$12\,288$	410×401	$1.02 \times 10^{+1}$	3.37×10^{-04}	-	-
$3 \times 3 \times 3$	$41\,472$	464×475	$3.43 \times 10^{+1}$	4.81×10^{-04}	3.4	6.2
$4 \times 4 \times 4$	$98\,304$	483×532	$7.92 \times 10^{+1}$	1.57×10^{-04}	2.3	3.7
$6 \times 6 \times 6$	$331\,776$	504×707	$2.96 \times 10^{+2}$	7.03×10^{-04}	3.7	6.2
$8 \times 8 \times 8$	$786\,432$	513×1014	$6.70 \times 10^{+2}$	4.70×10^{-04}	2.3	3.7
$10 \times 10 \times 10$	$1\,536\,000$	518×1502	$2.46 \times 10^{+3}$	3.53×10^{-04}	3.7	2.7

Numerical example — BIE on surfaces in 3D (2013, with J. Bremer and A. Gillman)

The local truncation error is set to 10^{-3}.

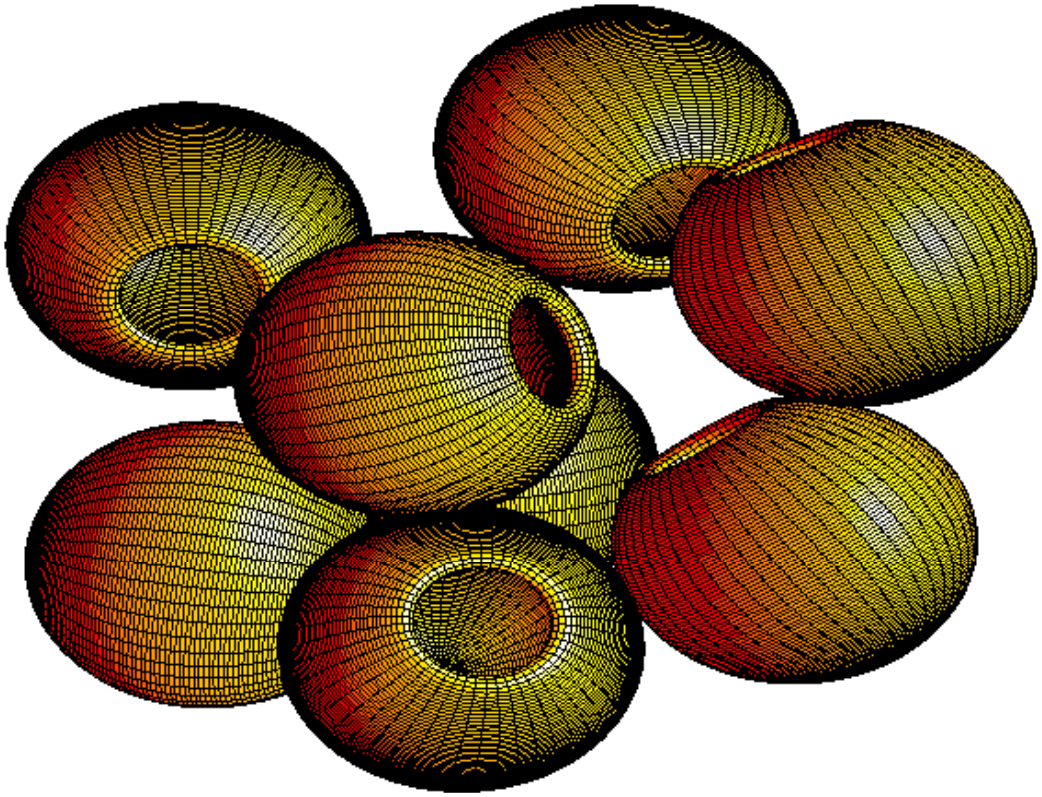
Grid dimensions	N	$N_{\text{out}} \times N_{\text{in}}$	T	E	Ratio	Predicted
$2 \times 2 \times 2$	$12\,288$	410×401	$1.02 \times 10^{+1}$	3.37×10^{-04}	-	-
$3 \times 3 \times 3$	$41\,472$	464×475	$3.43 \times 10^{+1}$	4.81×10^{-04}	3.4	6.2
$4 \times 4 \times 4$	$98\,304$	483×532	$7.92 \times 10^{+1}$	1.57×10^{-04}	2.3	3.7
$6 \times 6 \times 6$	$331\,776$	504×707	$2.96 \times 10^{+2}$	7.03×10^{-04}	3.7	6.2
$8 \times 8 \times 8$	$786\,432$	513×1014	$6.70 \times 10^{+2}$	4.70×10^{-04}	2.3	3.7
$10 \times 10 \times 10$	$1\,536\,000$	518×1502	$2.46 \times 10^{+3}$	3.53×10^{-04}	3.7	2.7

Increasing the accuracy is possible, but comes at a cost.

Now the local truncation error is set to 10^{-6}.

Grid dimensions	N	$N_{\text{out}} \times N_{\text{in}}$	T	E	Ratio	Predicted
$2 \times 2 \times 2$	$49\,152$	601×584	$1.61 \times 10^{+2}$	1.22×10^{-07}	-	-
$3 \times 3 \times 3$	$165\,888$	676×677	$6.87 \times 10^{+2}$	4.92×10^{-07}	4.3	6.2
$4 \times 4 \times 4$	$393\,216$	703×747	$1.68 \times 10^{+3}$	5.31×10^{-07}	2.4	3.6
$6 \times 6 \times 6$	$1\,327\,104$	728×925	$6.66 \times 10^{+3}$	4.60×10^{-06}	4.0	6.2
$8 \times 8 \times 8$	$3\,145\,728$	742×1237	$1.59 \times 10^{+4}$	2.30×10^{-07}	2.4	3.6

Example — scattering on domain with cavities (with S. Hao and P. Young)



Acoustic scattering on the exterior domain.

Each bowl is about 5λ.

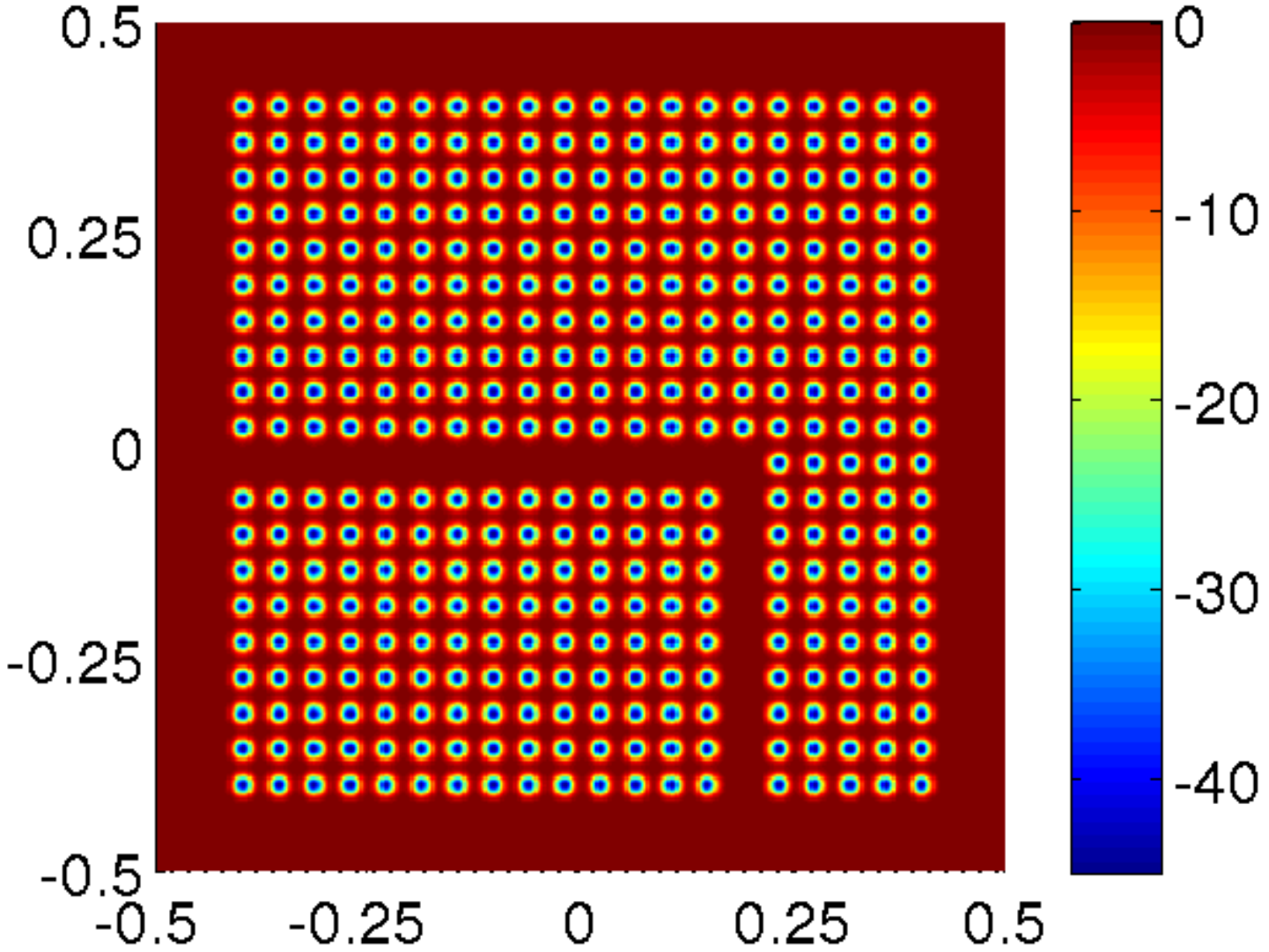
A hybrid direct/iterative solver is used (a highly accurate scattering matrix is computed for each body).

On an office desktop, we achieved an accuracy of 10^{-5}, in about 11h (all the time is spent in applying the inter-body interactions via the Fast Multipole Method).

Example: Free space scattering

$$\begin{cases} -\Delta u_{\text{out}}(\boldsymbol{x}) - k^2\,(1 - b(\boldsymbol{x}))\,u_{\text{out}}(\boldsymbol{x}) = -k^2\,b(\boldsymbol{x})\,u_{\text{in}}(\boldsymbol{x}) \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\big(\partial_{|\boldsymbol{x}|}u_{\text{out}}(\boldsymbol{x}) - ik\,u_{\text{out}}(\boldsymbol{x})\big) = 0 \end{cases}$$

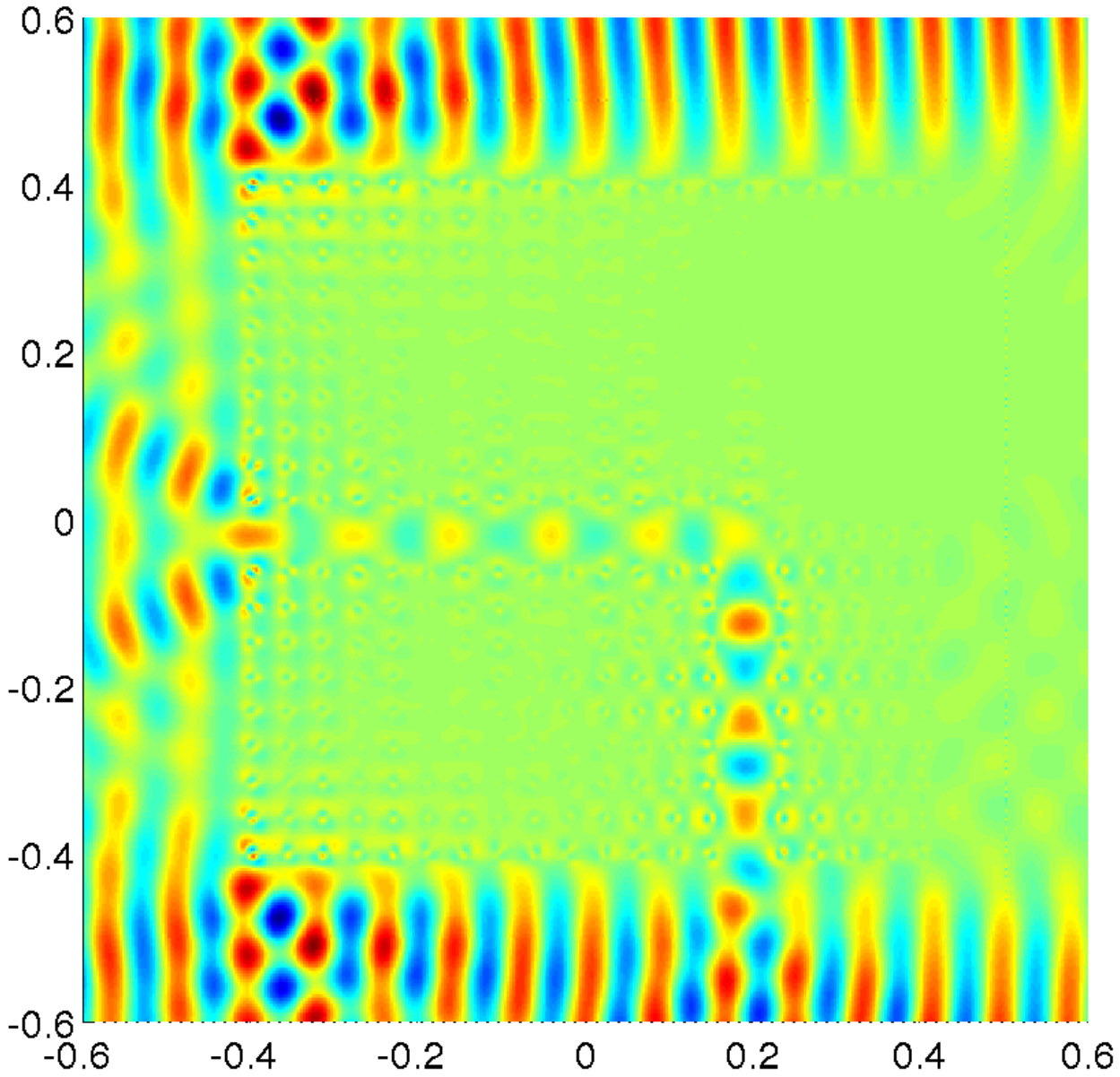
The scattering potential b — now a photonic crystal with a wave guide.



Example: Free space scattering

$$\begin{cases} -\Delta u_{\text{out}}(\boldsymbol{x}) - k^2\,(1 - b(\boldsymbol{x}))\,u_{\text{out}}(\boldsymbol{x}) = -k^2\,b(\boldsymbol{x})\,u_{\text{in}}(\boldsymbol{x}) \\ \lim_{|\boldsymbol{x}|\to\infty} \sqrt{|\boldsymbol{x}|}\big(\partial_{|\boldsymbol{x}|} u_{\text{out}}(\boldsymbol{x}) - ik\,u_{\text{out}}(\boldsymbol{x})\big) = 0 \end{cases}$$

The total field $u = u_{\text{in}} + u_{\text{out}}$ *(resulting from an incoming plane wave* $u_{\text{in}}(x) = \cos(k\,x_1)$*).*



CBMS conference (i.e. summer school):

"Fast direct solvers for elliptic PDEs"

Dartmouth College, June 23 - 27, 2014

Lecturer: Per-Gunnar Martinsson

Speakers: Alex Barnett
Adrianna Gillman
Leslie Greengard
Vladimir Rokhlin

Travel support available: Google "CBMS Fast Direct Solvers"

A broader view

The methods described follow a template:

1. *Develop a sketch of the data using randomized sampling.*

   The objective of this step is only to figure out "where to look"

   Randomized sampling introduces distortions, but for this stage, that is OK.
   E.g. "Johnson-Lindenstrauss projections" distort distances by factor $\frac{1+\varepsilon}{1-\varepsilon}$ using $O(\varepsilon^{-2})$ samples.

2. *Use deterministic method to compute a high-accuracy solution.*

   Once we know "where to look" we bring expensive but highly accurate deterministic methods to bear.

Claim: This template can be used in other contexts.

Example of application: Nearest neighbor search in $\mathbb{R}^D$

Peter Jones, Andrei Osipov, Vladimir Rokhlin

Suppose you are given N points $\{\boldsymbol{x}_j\}_{j=1}^N$ in $\mathbb{R}^D$. The coordinate matrix is

$$\mathbf{X} = [\boldsymbol{x}_1\ \boldsymbol{x}_2\ \cdots \boldsymbol{x}_N] \in \mathbb{R}^{D\times N}.$$

How do you find the k nearest neighbors for every point?

If $D = 2$, there are many options; you can, e.g, sort the points into a quad-tree. Similar things can be done for D small, but standard techniques get expensive fast as D grows.

Simple idea: Use a random map to project onto low-dimensional space. This "sort of" preserves distances. Execute a fast search there.

Improved idea: The output from a single random projection is unreliable. But, you can repeat the experiment several times, use these to generate a list of *candidates* for the nearest neighbors, and then compute exact distances to find the k closest among the candidates.

Final remarks:

- For large scale SVD/PCA of dense matrices, these algorithms are highly recommended; they compare favorably to existing methods in almost every regard. Free software can be downloaded $\rightarrow$ google *Mark Tygert*.

- The approximation error is a random variable, but its distribution is very tightly concentrated. Rigorous error bounds that are satisfied with probability $1 - \eta$ where η is a user set "failure probability" (*e.g.* $\eta = 10^{-10}$ or 10^{-20}).

- This talk mentioned *error estimators* only briefly, but they are important.
  Can operate independently of the algorithm for improved robustness.
  Typically cheap and easy to implement. Used to determine the actual rank.

- The theory can be hard, but *experimentation is easy!*
  Concentration of measure makes the algorithms behave as if deterministic.

- Randomized methods for computing "FMM"-style (HSS, $\mathcal{H}$-matrix, ...) representations of matrices have been developed — [M— 2008, 2011, 2014], [Lin, Lu, Ying 2011].

  Leads to accelerated (often $O(N)$) *direct solvers* for elliptic PDEs.
  Applications to scattering, composite materials, engineering design, etc.

- Survey paper: Google "Halko Martinsson Tropp"
  Slides: Under "Talks" tab on my webpage (google "Gunnar Martinsson").

- N. Halko, P.G. Martinsson, J. Tropp, “Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions.” *SIAM Review*, **53**(2), pp. 217–288, 2011.

- P.G. Martinsson, “A fast randomized algorithm for computing a Hierarchically Semi-Separable representation of a matrix”. *SIAM J. on Matrix Analysis and Appl.*, **32**(4), pp. 1251–1274, 2011.

- N. Halko, P.G. Martinsson, Y. Shkolnisky, M. Tygert, “An Algorithm for the Principal Component Analysis of large Data Sets”. *SIAM J. on Scientific Computation*, **33**(5), pp. 2580–2594, 2011.

- P.G. Martinsson, V. Rokhlin, and M. Tygert, “A randomized algorithm for the decomposition of matrices”. *Applied and Computational Harmonic Analysis*, **30**(1), pp. 47–68, 2011.

- E. Liberty, F. Woolfe, P.G. Martinsson, V. Rokhlin, and M. Tygert, “Randomized algorithms for the low-rank approximation of matrices”. *Proc. of the National Academy of Sciences*, **104**(51), 2007.

- H. Cheng, Z. Gimbutas, P.G. Martinsson, V. Rokhlin, “On the compression of low rank matrices”. *SIAM Journal of Scientific Computing*, **26**(4), pp. 1389-1404, 2005.