Modified from Talk in 2012
For RNLA study group, March 2015

Ming Gu, UC Berkeley
mgu@math.berkeley.edu

Low-Rank Approximations, Random Sampling and Subspace Iteration

Content

- Approaches for low-rank matrix approximations
- Random sampling and Subspace iteration
- Numerical experiments
- Future work

Low-Rank Matrix Approximation:

Problem Statement:

Given: mxn matrix A, and 0 <k < min(m,n) = n.

Goal: Compute a rank-k *approximation* to A.

- Fast low-rank matrix approximation is key to efficiency of superfast direct solvers for integral equations and many large sparse linear systems.
- Indispensable tool in mining large data sets.
- Randomized algorithms compute accurate truncated SVD.
- Minimum work and communication/Exceptionally high success rate.

Low-Rank Matrix Approximations: Current Approaches

More reliable

Lower cost

Modified Gram-Schmidt with column pivoting.

(not reliable)

Rank-Revealing QR factorization.

(guaranteed but limited reliability)

Partial SVD.

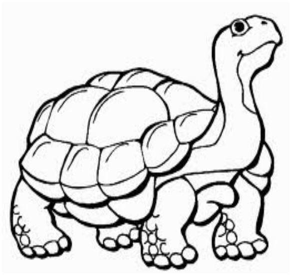
(limited reliability)

The Lanczos Algorithm.

(too much communication)

Truncated SVD

(Best quality, but too slow)



Low-Rank Matrix Approximations: Current Approaches

More reliable

Modified Gram-Schmidt with column pivoting.

(not reliable)

Rank-Revealing QR/LU factorization.

(**Actually better than random sampling. Talks in April**)

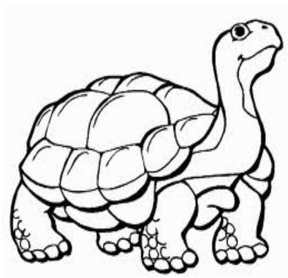
Lower cost

Partial SVD.

(limited reliability)

The Lanczos Algorithm.

(too much communication)



Truncated SVD

(Best quality, but too slow)

Low-Rank Matrix Approximations: Goals

Highly Efficient

Minimum communication

As accurate/reliable as Truncated SVD

Golden Standard: Truncated SVD

Given mxn matrix with n ≤ m, the SVD of A is

$$A = U\,\Sigma V^T = (u_1 \quad \cdots \quad u_n)\begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix}(v_1 \quad \cdots \quad v_n)^T$$

The rank-k truncated SVD is

$$A_k = U_k \Sigma_k V_k^T = (u_1 \quad \cdots \quad u_k)\begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{pmatrix}(v_1 \quad \cdots \quad v_k)^T$$

Theorem [Can't beat A_k] (Eckart & Young, 1936)

$$Min_{rank(B)\le k} \| A - B \|_2 = \| A - A_k \|_2 = \sigma_{k+1}$$

$$Min_{rank(B)\le k} \| A - B \|_F = \| A - A_k \|_F = \sqrt{\sum_{j=k+1}^{n} \sigma_j^2}$$

Low-rank Approximations: Strong Rank-Revealing QR

Theorem [Limited Warranty] (Gu & Eisenstat, 1994)

Given mxn matrix with n ≤ m, there exists a permutation Π,

$$A\,\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix} \; with \;\; 1 \le {}^{\sigma_j}\!/_{\sigma_j(R_{11})} \le \sqrt{1 + 4k(n-k)}$$

$$j = 1, \cdots, k.$$

- Factorization can be computed in O(mnk) operations.

- Basis for some popular low-rank matrix approximation schemes.

- Permutation can be arbitrary, excessive communication possible.

Low-rank Approximations: Strong Rank-Revealing QR

Theorem [Limited Warranty] (Gu & Eisenstat, 1994)

Given mxn matrix with n ≤ m, there exists a permutation Π,

$$\mathrm{A}\,\Pi = \mathrm{Q}\begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix} \; with \;\; 1 \le \sigma_j / \sigma_j(R_{11}) \le \sqrt{1 + 4k(n-k)}$$

$$j = 1, \cdots, k.$$

- ~~Factorization can be computed in O(mnk) operations.~~ 
- Basis for some popular low-rank matrix approximation schemes. 
- ~~Permutation can be arbitrary, excessive communication possible.~~ 

Low-Rank Approximations: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

  1. Draw a random nx(k+p) matrix Ω.
  2. Compute QR = A Ω
  3. and SVD: $Q^T A = \hat{U}\, \hat{\Sigma} \hat{V}^T$
  4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$

Low-Rank Approximation: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$

- Easy to implement.
- Very efficient computation.
- Minimum communication.

Low-Rank Approximation: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$

- Easy to implement.
- Very efficient computation.
- Minimum communication.

Thm [(Remarkable) Limited Warranty]

(Halko/Martinsson/Tropp, 2011)

$$\| A - B \|_2 = O(\sigma_{k+1}) \gg \sigma_{k+1}$$

with failure probability $5p^{-p}$

Low-Rank Approximation: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k\hat{\Sigma}_k\hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$

- Easy to implement.
- Very efficient computation.
- Minimum communication.

Thm [(Remarkable) Limited Warranty]

(Halko/Martinsson/Tropp, 2011)

$$\| A - B \|_2 = O(\sigma_{k+1}) \gg \sigma_{k+1}$$

with failure probability $5p^{-p}$

4 lines of code

40 pages of analysis

Low-Rank Approximation: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$

- Easy to implement.
- Very efficient computation.
- Minimum communication.

Thm [(Remarkable) Limited Warranty]

(Halko/Martinsson/Tropp, 2011)

$$\| A - B \|_2 = O(\sigma_{k+1}) \gg \sigma_{k+1}$$

with failure probability $5p^{-p}$

(For p = 13, $5p^{-p} = 1.6 * 10^{-14}$)

Is 10^{-14} small enough failure chance?

Chance of DNA match = 10^{-14}

For the Truly Motivated, Bound in Full Glory

$$\| A - B \|_2 \leq \sigma_{k+1} + \| (I - P_Y)A \|_2$$

THEOREM 10.8 (Deviation bounds for the spectral error). *Frame the hypotheses of Theorem 10.5. Assume further that $p \geq 4$. For all $u, t \geq 1$,*

$$\begin{aligned} &\|(\mathbf{I} - \boldsymbol{P_Y})\boldsymbol{A}\| \\ &\quad \leq \left[\left(1 + t \cdot \sqrt{12k/p}\right)\sigma_{k+1} + t \cdot \frac{\mathrm{e}\sqrt{k+p}}{p+1}\left(\sum\nolimits_{j>k}\sigma_j^2\right)^{1/2}\right] + ut \cdot \frac{\mathrm{e}\sqrt{k+p}}{p+1}\sigma_{k+1}, \end{aligned}$$

with failure probability at most $5t^{-p} + \mathrm{e}^{-u^2/2}$.

Low-Rank Matrix Approximations: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$

- Earlier work by Rokhlin/Tygert (2008)
- Many variations.
- Randomized rank-revealing QR (Demmel/Dumitriu/Holtz, 2008; Toledo, 2010)

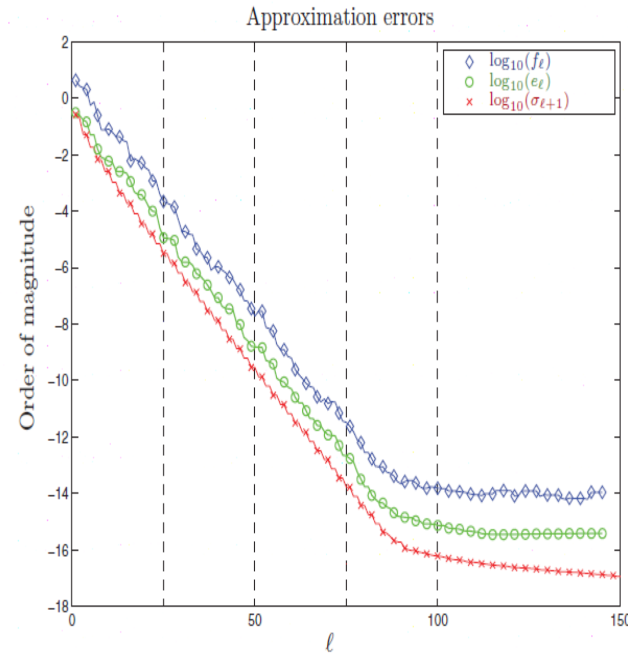
Low-Rank Approximation: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$



- Algorithm can work far better than theory predicts.

Improved Randomized Sampling

Algorithm RandSam1

- Input: mxn matrix A, int k, p, c.

1. Draw a random nx(k+p+c) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$

Only change from RandSam0:
p becomes p + c

Smallest modification of any algorithm.

Improved Randomized Sampling

Algorithm RandSam1

- Input: mxn matrix A, int k, p, c.

1. Draw a random nx(k+p+c) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U}\, \hat{\Sigma} \hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$

Only change from RandSam0:
p becomes p + c

Smallest modification of any algorithm.

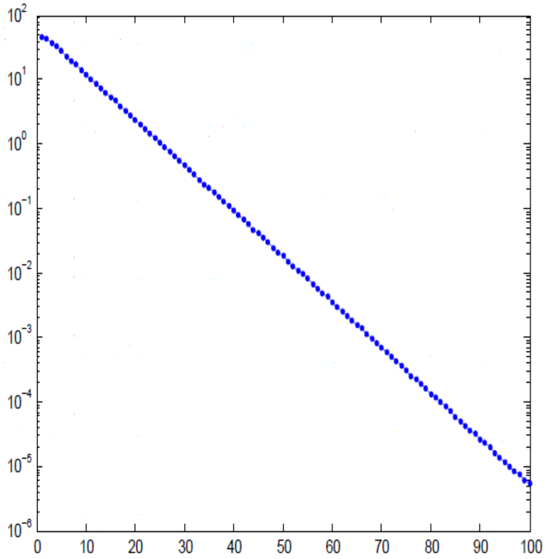
c allows a drastically different error bound, controls accuracy.

p remains in control of failure chance.

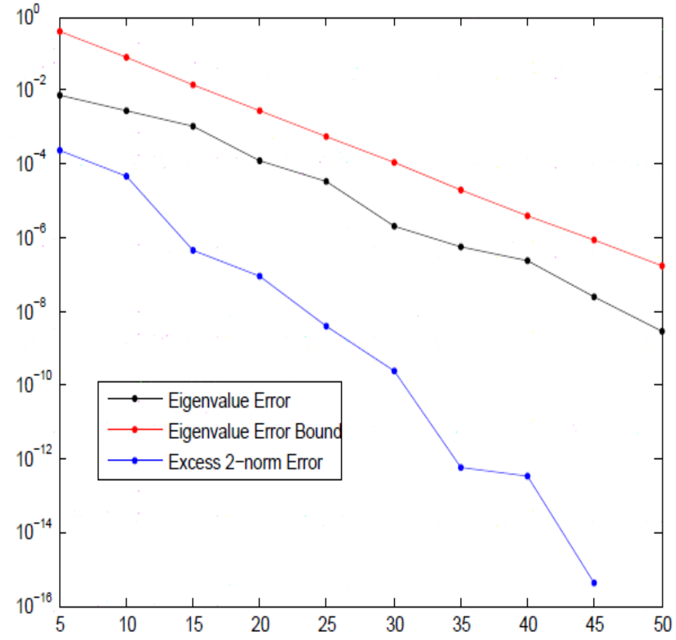
Improved Randomized Sampling

A = 2000x2000 random matrix with geometrically decaying singular values.

p = 10, k = 20. $\sigma_1 = 45$, $\sigma_{20} = 2$



Accuracy increases with c



Randomized Power Method (I)

Algorithm RandSam2

- Input: mxn matrix A, int k, p, c, q.

1. Draw a random nx(k+p+c) matrix Ω.
2. Compute QR of $(AA^T)^q A \Omega$
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k\hat{\Sigma}_k\hat{V}_k^T$

- Output:

$$B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$$

QR needs done carefully for numerical accuracy.

Algorithm is old one when q = 0;
but q = 1 far more accurate.

Should converge faster when singular values do not decay very fast.

Thm [Limited Warranty]
(Halko/Martinsson/Tropp, 2011)

$$\| A - B \|_2 = O(\sigma_{k+1}) > \sigma_{k+1}$$

with failure probability $5p^{-p}$

Randomized Power Method (I)

Algorithm RandSam2

- Input: mxn matrix A, int k, p, c, q.

1. Draw a random nx(k+p+c) matrix Ω.
2. Compute QR of $(AA^T)^q A \Omega$
3. and SVD: $Q^T A = \hat{U}\,\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output:

$$B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$$

Traditional Subspace Iteration with random start matrix

QR needs done carefully for numerical accuracy.

Algorithm is old one when q = 0;
but q = 1 far more accurate.

Should converge faster when singular values do not decay very fast.

Thm [Limited Warranty]
(Halko/Martinsson/Tropp, 2011)

$$\| A - B \|_2 = O(\sigma_{k+1}) > \sigma_{k+1}$$

with failure probability $5p^{-p}$

Bounds both stronger and weaker than those for traditional Subspace Iteration

New Error Bound Analysis

THEOREM 5.8. *Let $A = U\Sigma V^T$ be the SVD of A, and $0 \leq p \leq \ell - k$. Further let QB_k be a rank-k approximation computed by Algorithm 2.2. Given any $0 < \Delta \ll 1$, define*

$$\mathcal{C}_\Delta = \frac{e\sqrt{\ell}}{p+1}\left(\frac{2}{\Delta}\right)^{\frac{1}{p+1}}\left(\sqrt{n-\ell+p}+\sqrt{\ell}+\sqrt{2\log\frac{2}{\Delta}}\right).$$

We must have for $j = 1, \cdots, k$,

$$\sigma_j\left(QB_k\right) \geq \frac{\sigma_j}{\sqrt{1+\mathcal{C}_\Delta^2\left(\frac{\sigma_{\ell-p+1}}{\sigma_j}\right)^{4q+2}}},$$

and

$$\|\left(I-QQ^T\right)A\|_F \leq \|A-QB_k\|_F \leq \sqrt{\left(\sum_{j=k+1}^{n}\sigma_j^2\right)+k\mathcal{C}_\Delta^2\sigma_{\ell-p+1}^2\left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{4q}},$$

$$\|\left(I-QQ^T\right)A\|_2 \leq \|A-QB_k\|_2 \leq \sqrt{\sigma_{k+1}^2+k\mathcal{C}_\Delta^2\sigma_{\ell-p+1}^2\left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{4q}}.$$

with exception probability at most Δ.

Fast Randomized Algorithm with

Subsampled random Fourier Transform: $T_{\text{struct}} \sim mn\log(\ell) + \ell^2 n$

HALKO, MARTINSSON, AND TROPP

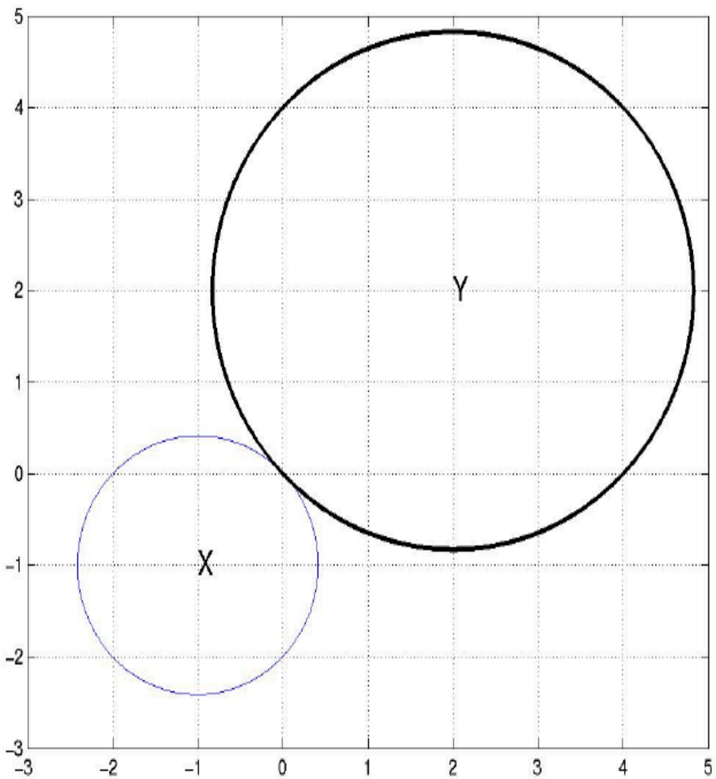
ALGORITHM 4.5: FAST RANDOMIZED RANGE FINDER
Given an $m \times n$ matrix $\boldsymbol{A}$, and an integer ℓ, this scheme computes an $m \times \ell$ orthonormal matrix $\boldsymbol{Q}$ whose range approximates the range of $\boldsymbol{A}$.
1 Draw an $n \times \ell$ SRFT test matrix $\boldsymbol{\Omega}$, as defined by (4.6).
2 Form the $m \times \ell$ matrix $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{\Omega}$ using a (subsampled) FFT.
3 Construct an $m \times \ell$ matrix $\boldsymbol{Q}$ whose columns form an orthonormal basis for the range of $\boldsymbol{Y}$, e.g., using the QR factorization $\boldsymbol{Y} = \boldsymbol{Q}\boldsymbol{R}$.

Numerical Experiment (I): Computing truncated SVD

Comparison between randomized algorithm and svds

$$A = (\log\ |x_i - y_j|)$$ is 4000x4000 matrix,

x_i, y_j *disjoint 2D points*



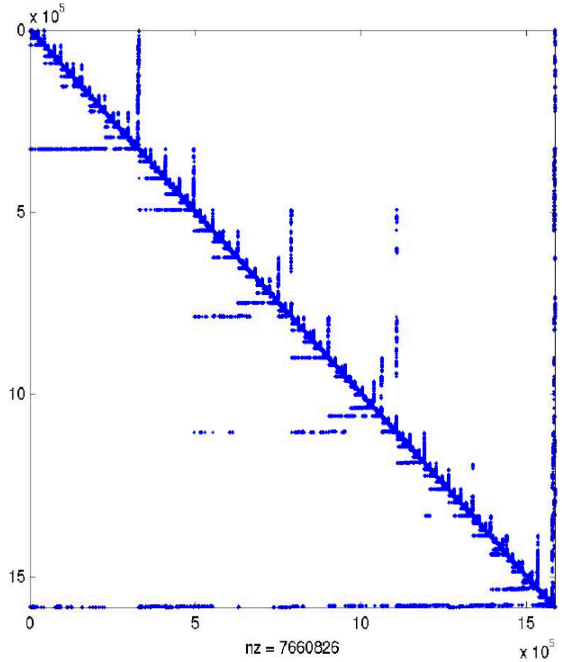
Numbers of Matrix-Vector Multiplies

Tolerance	$q=0$	$q=2$	$q=4$	svds
10^{-6}	143	5×96	9×79	500
10^{-8}	180	5×96	9×87	600
10^{-10}	190	5×96	9×93	600

Numerical Experiment (II):

Fast Structured Matrix Preconditioners:

T. Davis' SPD Sparse Matrix



A = Bottom Schur Complement of dimension 3300.
CG takes 878 iterations for 10^{-12} residual

Numbers of PCG Iterations

Maximum off-diagonal rank k	$p = 10$	$p = 20$	$p = 40$
20	75	77	72
40	69	69	69
60	64	61	61

Numerical Experiment (III): Eigenface

Comparison with truncated SVD for 200 classifications

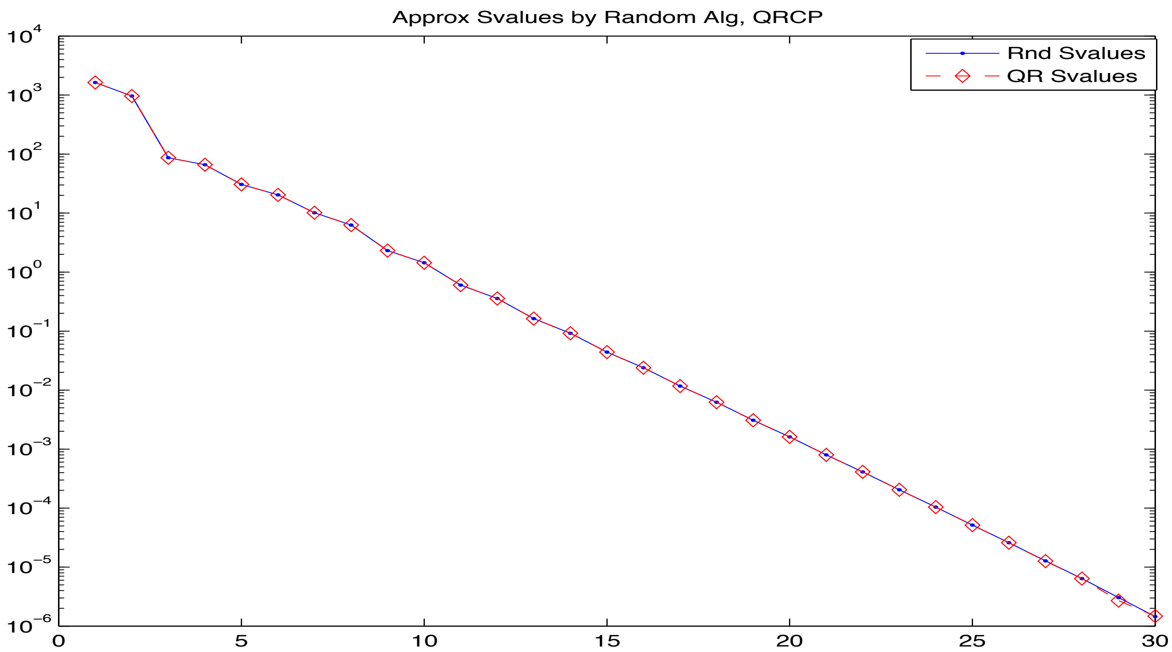
Comparison of Numbers of Incorrect Matches

Rank k	$p = 10$	$p = 20$	$p = 40$	Truncated SVD
10	32	25	23	24
20	25	26	25	21
30	21	20	18	17
40	20	17	17	16

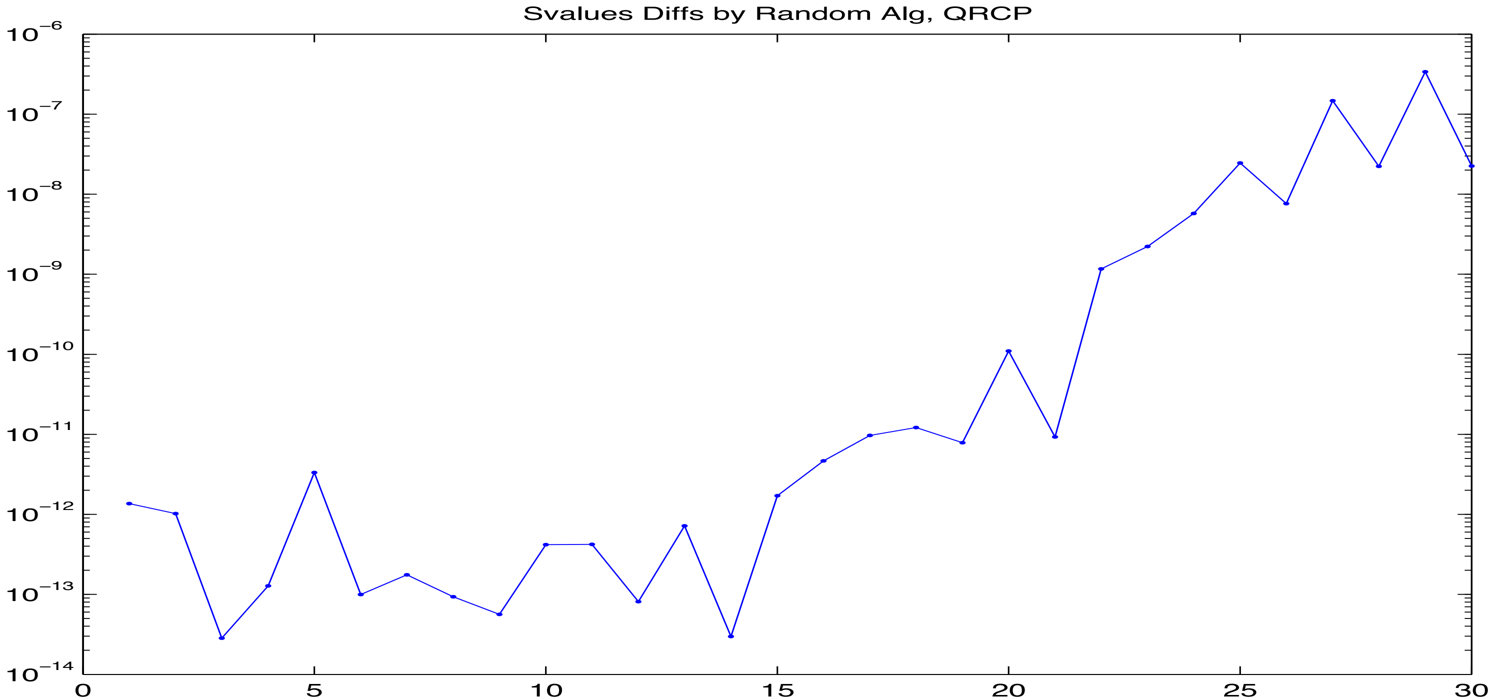
Numerical Experiment (IV): QRCP vs. Randomized Algorithm

```
N = 4000;
x = (pi/2)*(-N:N)/N;
e = ones(2*N+1,1);
B = log((abs(sin(e*x-x'*e'))));
B(~isfinite(B)) = 0;
B=B(1:N/2,N/2+1:end);

p = 30;
W = randn(N+N/2+1,p);
BW = B*W;
[QW,RW]=qr(BW,0);
[UW,SW,VW] = svd(B'*QW,0);
srnd = diag(SW);

[QB,RB,P] = qr(B,0);
sqr = svd(RB(1:p,:));
```

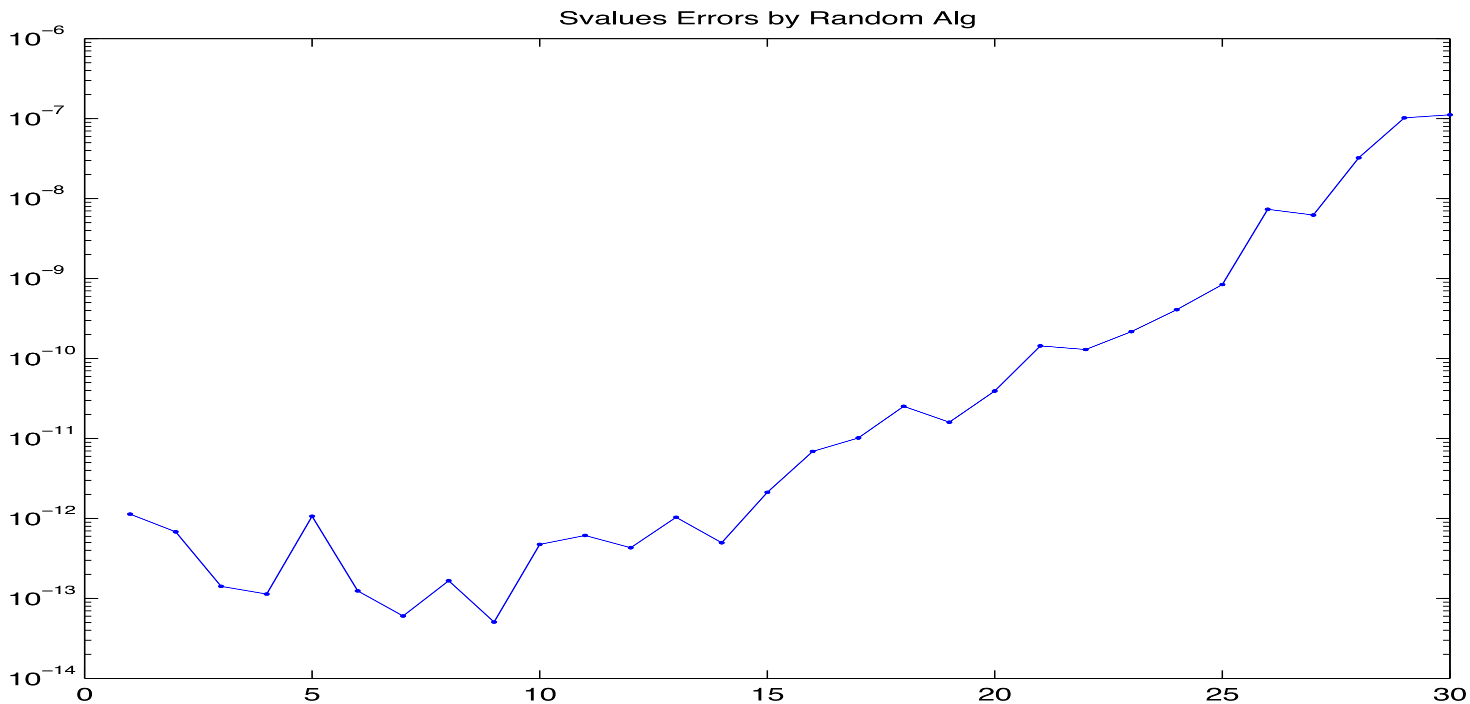
Numerical Experiment (IV): QRCP vs. Randomized Algorithm



Numerical Experiment (IV): QRCP vs. Randomized Algorithm



Numerical Experiment (IV): QRCP vs. Randomized Algorithm



Current Work On Randomized Algorithms

- Randomized Gaussian elimination with complete pivoting

- Randomized spectrum-revealing LU, QR, Cholesky factorizations

  (vs. CUR/CX decompositions,
  pivoted Cholesky factorizations,
  randomized low-rank matrix approximations)

- Big Data applications.