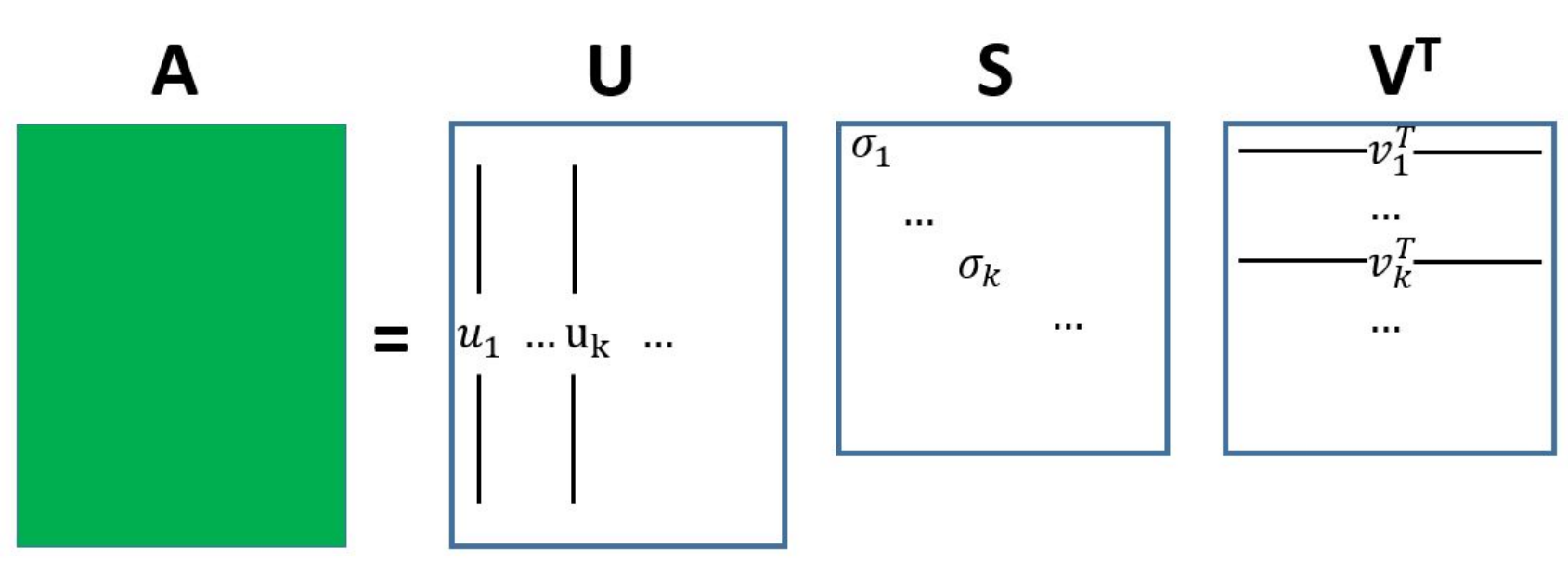
RANDOMISED ALGORITHMS FOR LOW-RANK APPROXIMATION

Kaspar Märtens Sherman Ip

6 November 2015

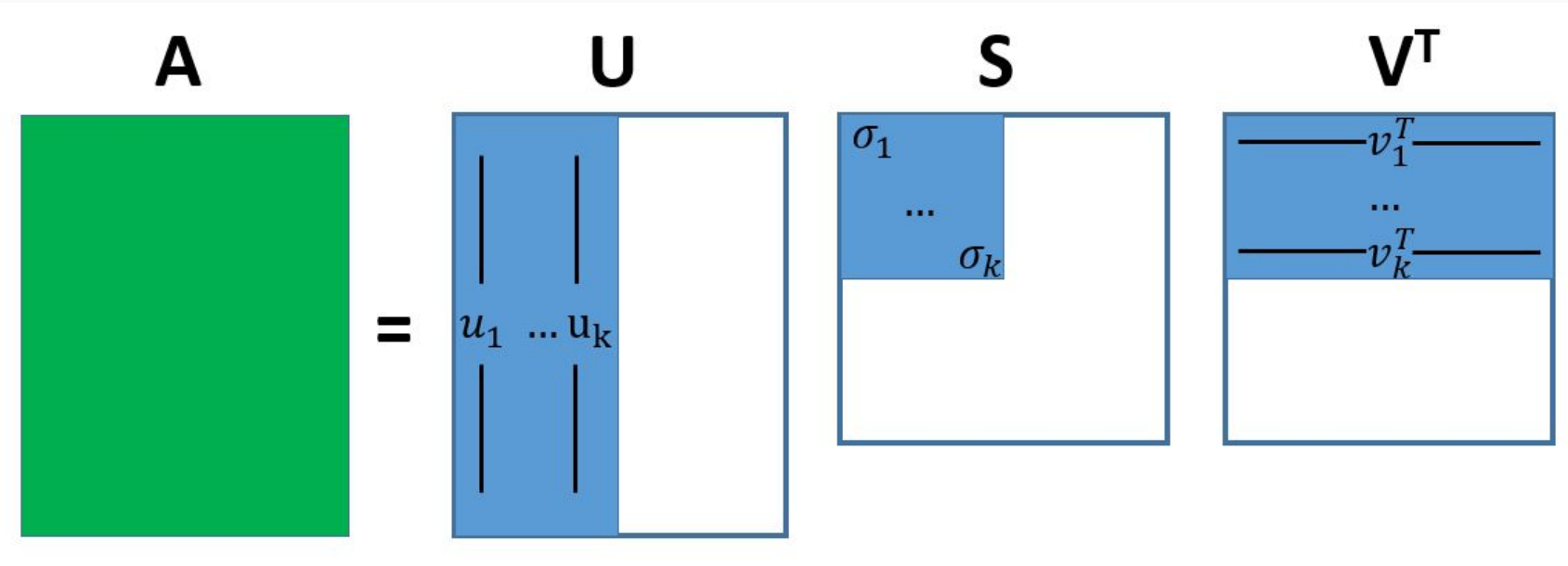
SVD

Singular value decomposition for matrix $A \in \mathbb{R}^{m,n}$



Truncated SVD

Singular value decomposition for matrix $A \in \mathbb{R}^{m,n}$



$$\tilde{A}_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

TRUNCATED SVD ILLUSTRATION

$k \in \{3, 5, 10, 30, 50, 100\}$

RANDOMISED ALGORITHM

Randomised algorithm

Idea: find a rank k approximation such that

$$A \approx QQ^{\mathrm{T}} A \tag{1}$$

Randomised algorithm

Idea: find a rank k approximation such that

$$A \approx QQ^{\mathrm{T}} A \tag{1}$$

where Q is a $m \times k$ matrix with orthonormal columns $\phi_1, \phi_2, \ldots, \phi_k$

$$Q = \begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ \phi_1 & \phi_2 & \cdots & \phi_k \\ \downarrow & \downarrow & & \downarrow \end{pmatrix} \tag{2}$$

Randomised algorithm

Idea: find a rank k approximation such that

$$A \approx QQ^{\mathrm{T}} A \tag{1}$$

where Q is a $m \times k$ matrix with orthonormal columns $\phi_1, \phi_2, \ldots, \phi_k$

$$Q = \begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ \phi_1 & \phi_2 & \cdots & \phi_k \\ \downarrow & \downarrow & & \downarrow \end{pmatrix} \tag{2}$$

so that

$$QQ^T = \sum_{i=1}^{k} \phi_i \phi_i^{\mathrm{T}} \tag{3}$$

is a projector.

Randomised algorithm

Idea: find a rank k approximation such that

$$A \approx QQ^{\mathrm{T}} A \tag{1}$$

where Q is a $m \times k$ matrix with orthonormal columns $\phi_1, \phi_2, \ldots, \phi_k$

$$Q = \begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ \phi_1 & \phi_2 & \cdots & \phi_k \\ \downarrow & \downarrow & & \downarrow \end{pmatrix} \tag{2}$$

so that

$$QQ^T = \sum_{i=1}^{k} \phi_i \phi_i^{\mathrm{T}} \tag{3}$$

is a projector.

We want the orthonormal columns to span the range of A.

Bases in the range of A

To construct orthonormal bases which span the range of A, we can sample the range of A,

$$\text{range}(A) = \{Ax : \ x \in \mathbb{R}^n\}$$

Bases in the range of A

To construct orthonormal bases which span the range of A, we can sample the range of A,

$$\text{range}(A) = \{Ax : \ x \in \mathbb{R}^n\}$$

Sample $\omega_i \sim \text{N}(0, I)$, then

$$y_i = A\omega_i \qquad \text{for i.i.d. } \ i = 1, 2, \ldots, k \tag{4}$$

are samples of the range of A.

Bases in the range of A

To construct orthonormal bases which span the range of A, we can sample the range of A,

$$\text{range}(A) = \{Ax : \ x \in \mathbb{R}^n\}$$

Sample $\omega_i \sim \text{N}(0, I)$, then

$$y_i = A\omega_i \qquad \text{for i.i.d. } \ i = 1, 2, \dots, k \tag{4}$$

are samples of the range of A.

Using $y_1, y_2, \dots, y_k$, construct orthonormal bases $\phi_1, \phi_2, \dots, \phi_k$ using existing algorithms such as MATLAB's `orth` function.

Bases in the range of A

To construct orthonormal bases which span the range of A, we can sample the range of A,

$$\text{range}(A) = \{Ax : \ x \in \mathbb{R}^n\}$$

Sample $\omega_i \sim \text{N}(0, I)$, then

$$y_i = A\omega_i \qquad \text{for i.i.d. } \ i = 1, 2, \dots, k \tag{4}$$

are samples of the range of A.

Using $y_1, y_2, \dots, y_k$, construct orthonormal bases $\phi_1, \phi_2, \dots, \phi_k$ using existing algorithms such as MATLAB's `orth` function.

It turns out that it is useful to replace k with $k + p$, $p \geq 1$.

Implementation

Putting all of this together, the algorithm only takes two lines in MATLAB

$$Q = \texttt{orth}(A \times \texttt{randn}(n, k + p))$$

Implementation

Putting all of this together, the algorithm only takes two lines in MATLAB

$$Q = \texttt{orth}(A \times \texttt{randn}(n, k+p))$$

$$A \approx QQ^{\mathrm{T}} A$$

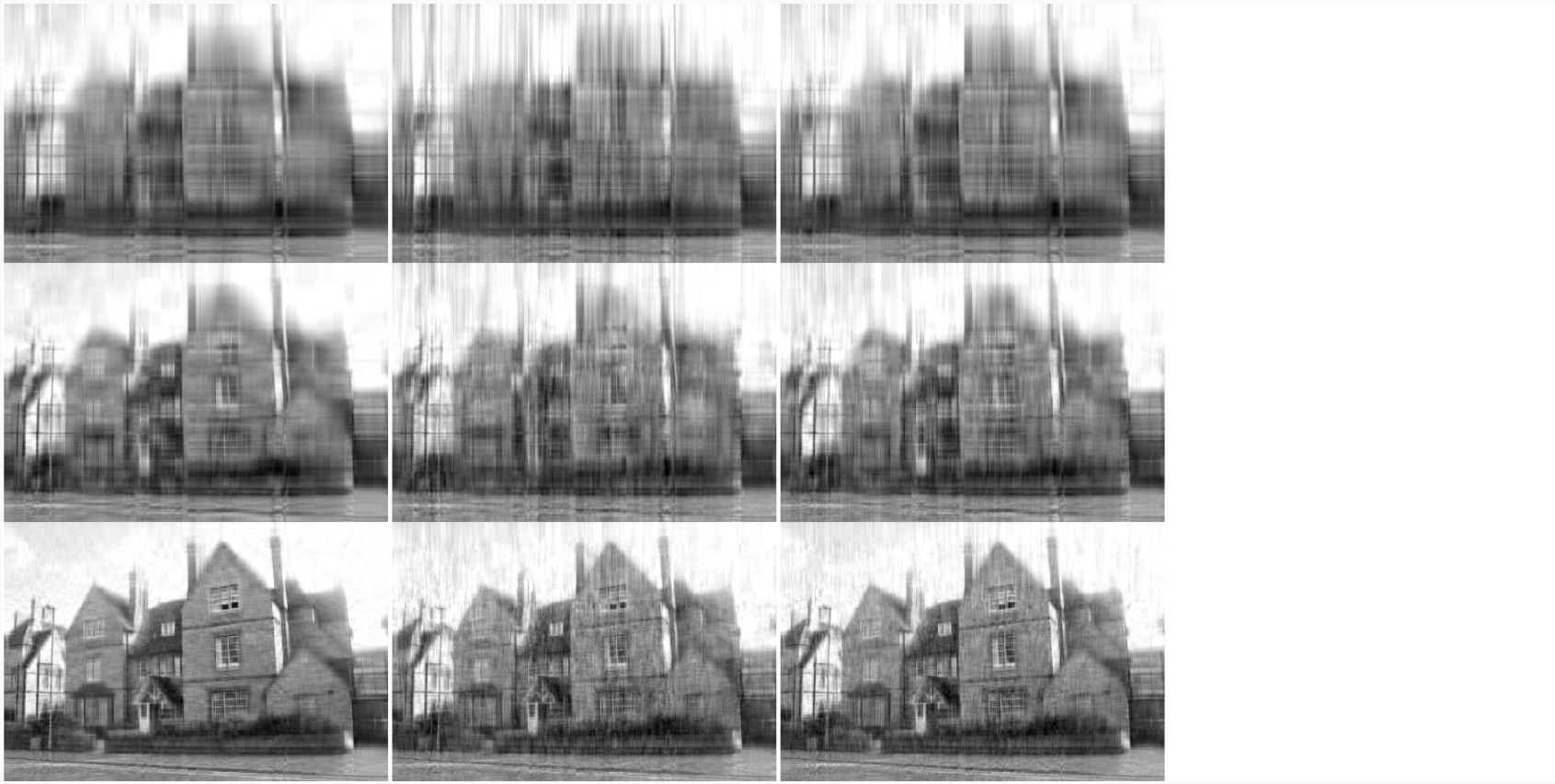
Approximate SVD

This gives us an algorithm for approximate SVD:

Construct Q

$$Q = \texttt{orth}(A \times \texttt{randn}(n, k+p))$$

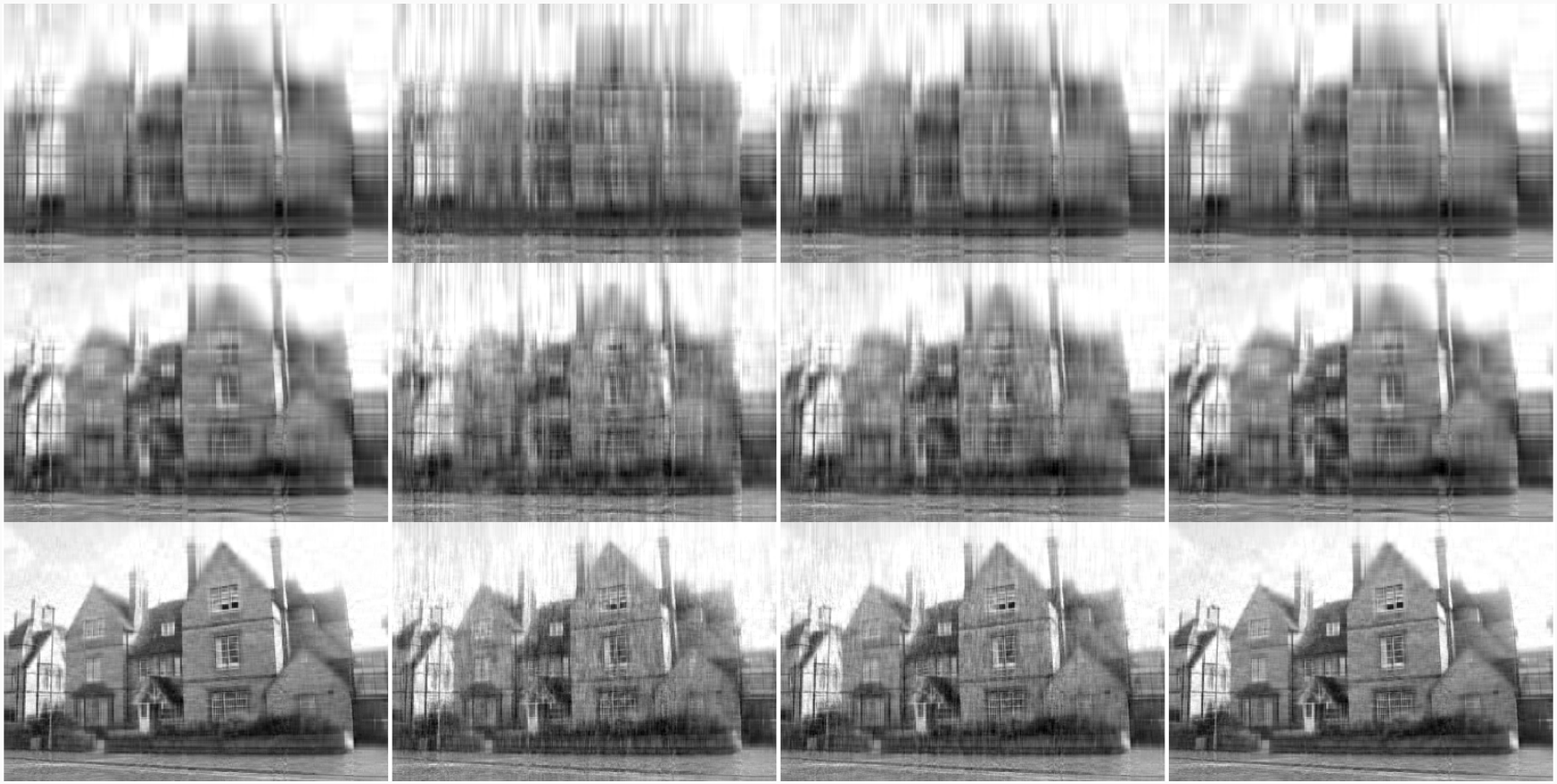
Instead of constructing SVD for $A \approx QQ^{\mathrm{T}}A$, obtain SVD for a small matrix $Q^{\mathrm{T}}A$.



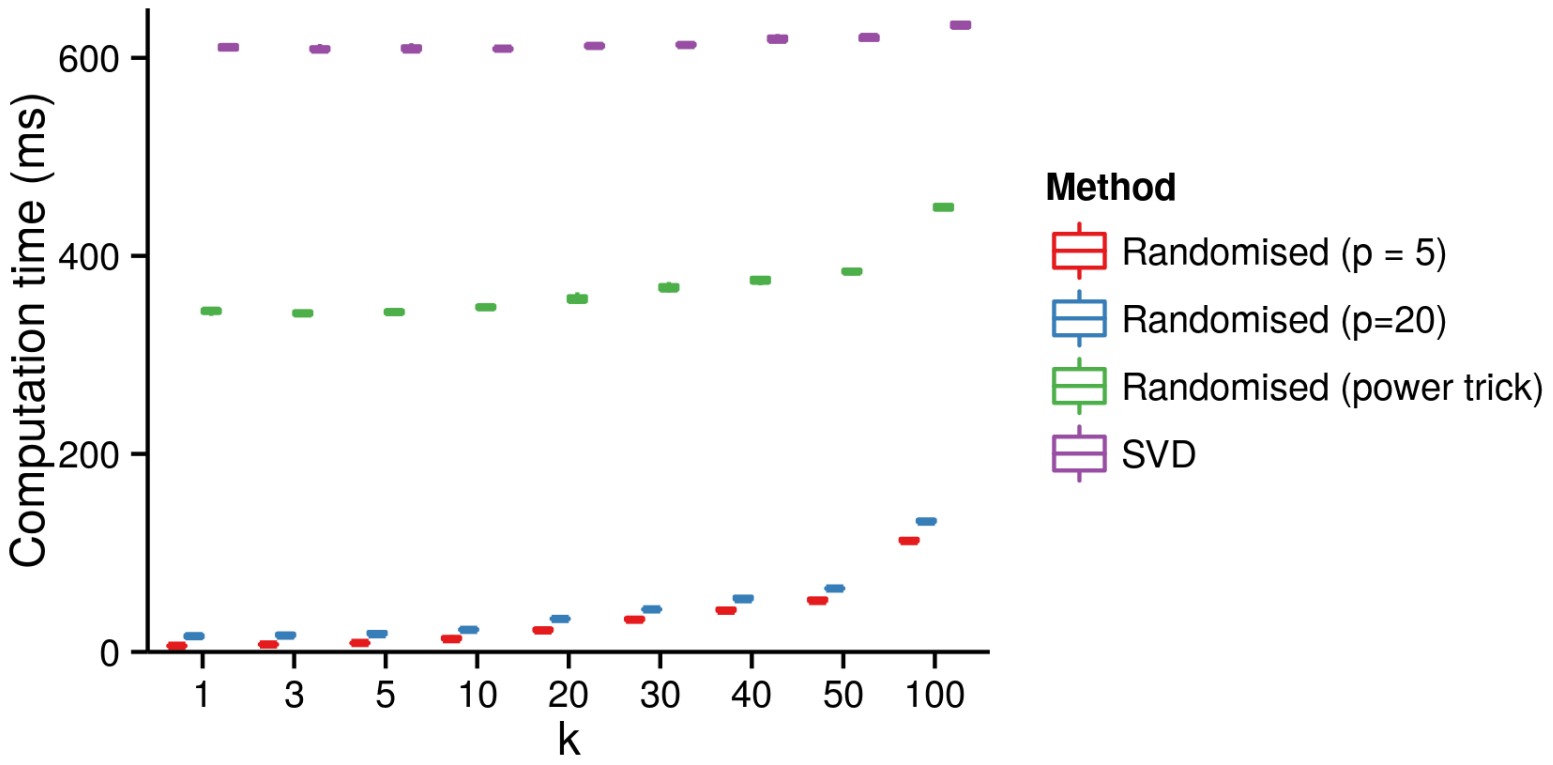
Power trick: improving approximate SVD

Power trick becomes useful when the singular values of A do not decay rapidly enough.

Idea: instead of approximating the range of A, approximate the range of $(AA^T)^q A$ for some small $q \in \{1, 2, 3\}$.



Computation times



QUANTIFYING APPROXIMATION ACCURACY

Residual spectral norm

Given a matrix A and low rank approximation $QQ^{\mathrm{T}}A$, the residual spectral norm can be taken

$$\|A - QQ^{\mathrm{T}}A\| \tag{5}$$

to quantify the approximation accuracy. This is the largest singular value of $\|A - QQ^{\mathrm{T}}A\|$

Residual spectral norm

Given a matrix A and low rank approximation $QQ^{\mathrm{T}}A$, the residual spectral norm can be taken

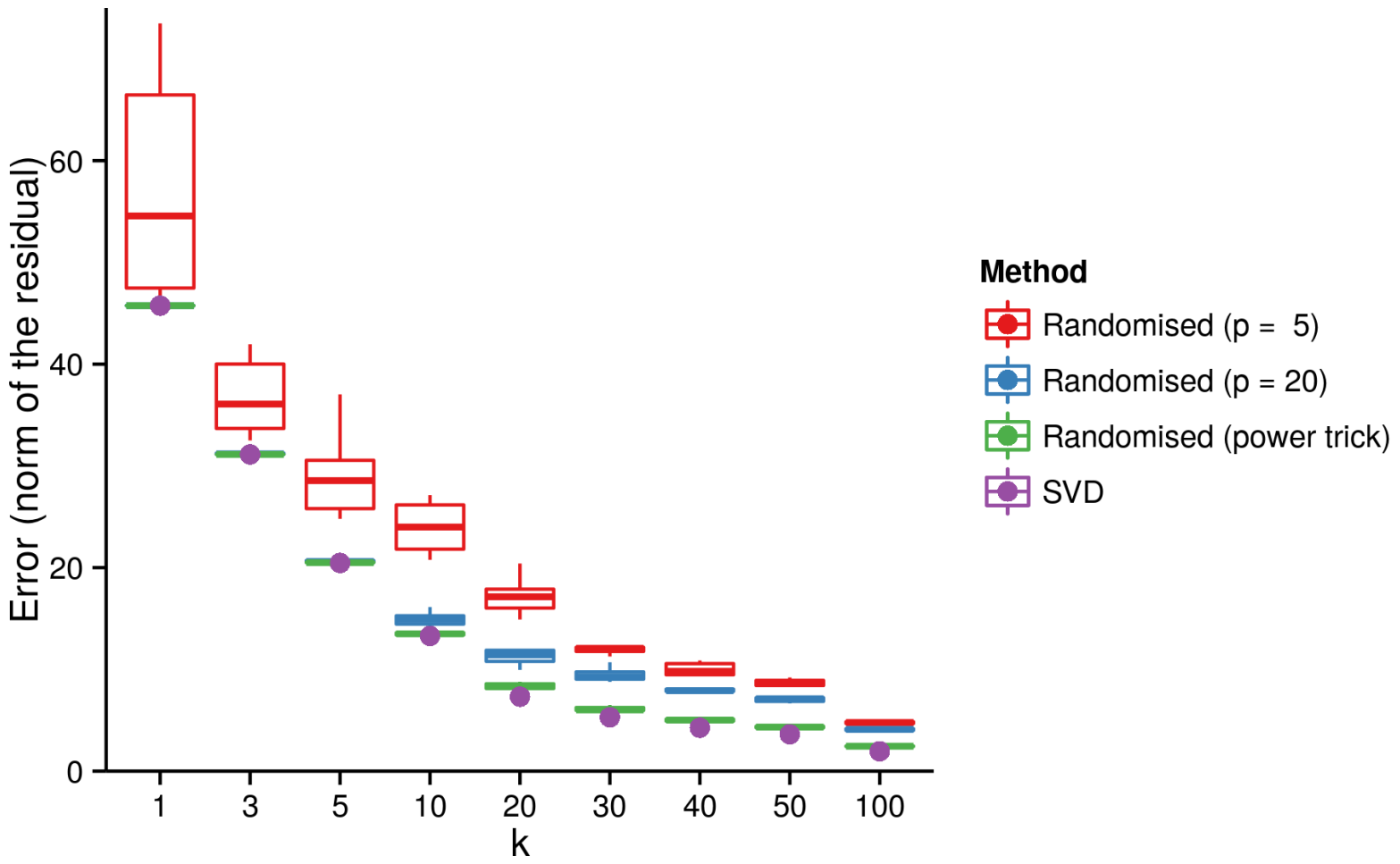
$$\|A - QQ^{\mathrm{T}}A\| \tag{5}$$

to quantify the approximation accuracy. This is the largest singular value of $\|A - QQ^{\mathrm{T}}A\|$

For exact SVD truncation, $\|A - QQ^{\mathrm{T}}A\| = \sigma_{k+1}$.

For the randomised approach, it is a random variable.

Observed $||A - QQ^{\mathsf{T}}A||$ for image example



Upper bound

Our goal is to bound $\mathbb{E}||A - QQ^{\mathrm{T}}A||$.

Upper bound

Our goal is to bound $\mathbb{E}||A - QQ^{\mathrm{T}}A||$.

For example, for a rank k approximation (i.e. when Q has k columns) can we find a constant c such that

$$\mathbb{E}||A - QQ^{\mathrm{T}}A|| \leq c\,\sigma_{k+1}\,?$$

Upper bound

Our goal is to bound $\mathbb{E}||A - QQ^{\mathrm{T}}A||$.

For example, for a rank k approximation (i.e. when Q has k columns) can we find a constant c such that

$$\mathbb{E}||A - QQ^{\mathrm{T}}A|| \leq c\,\sigma_{k+1}\,?$$

This is not possible. However, this constant exists for a rank $k+p$ approximation ($p \geq 1$).

Upper bound

N. Halko, P.G. Martinsson and J.A. Tropp (2011) introduced an upper bound on the expected residual spectral norm.

Upper bound

N. Halko, P.G. Martinsson and J.A. Tropp (2011) introduced an upper bound on the expected residual spectral norm.

$$\mathrm{E}\left[\|A - QQ^{\mathrm{T}}A\|\right] \leqslant \left[1 + \frac{4\sqrt{(k+p)\min(m,n)}}{p-1}\right]\sigma_{k+1} \qquad (6)$$

Worst case matrix

R. Witten and E. Candès (2015) improved the bounds by introducing a random variable W such that

$$||A - QQ^{\mathrm{T}} A|| \overset{d}{\leq} \sigma_{k+1} W \tag{7}$$

Worst case matrix

R. Witten and E. Candès (2015) improved the bounds by introducing a random variable W such that

$$||A - QQ^{\mathrm{T}}A|| \overset{d}{\leq} \sigma_{k+1}\, W \tag{7}$$

with expectation

$$\sup_{A} \mathbb{E}||A - QQ^{\mathrm{T}}A|| = \sigma_{k+1}\, \mathbb{E}\, W\,. \tag{8}$$

Worst case matrix

The expectation can be bounded. Defining $d = \min(m, n)$, the upper bound is given as

$$\sup_A \mathbb{E}||A - QQ^{\mathrm{T}}A|| \leq \sigma_{k+1}\left[1 + (\sqrt{d-k} + \sqrt{k})\, e\frac{\sqrt{k+p}}{p}\right] \tag{9}$$

Worst case matrix

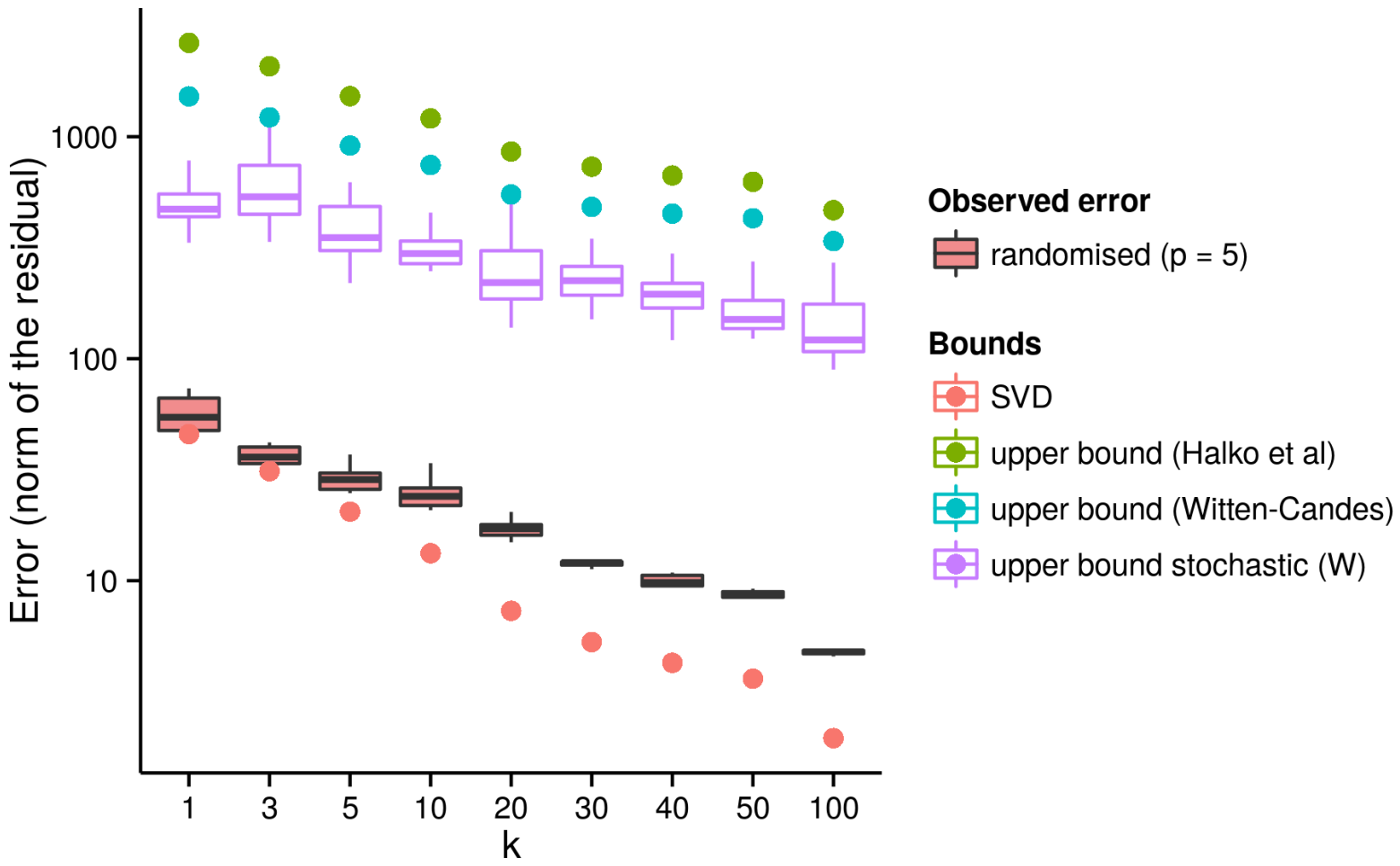
The expectation can be bounded. Defining $d = \min(m, n)$, the upper bound is given as

$$\sup_A \mathbb{E}||A - QQ^{\mathrm{T}}A|| \leq \sigma_{k+1}\left[1 + (\sqrt{d-k} + \sqrt{k})\, e\frac{\sqrt{k+p}}{p}\right] \tag{9}$$

and the lower bound as

$$\sup_A \mathbb{E}||A - QQ^{\mathrm{T}}A|| \geq \sigma_{k+1}\sqrt{\frac{d-(k+p+2)}{p+1}} \tag{10}$$

Upper bounds for the image example



SOME MORE EXPERIMENTS

Simulating matrices

Recall a matrix A can be decomposed using SVD

$$A = USV^{\mathrm{T}} \ . \tag{11}$$

Simulating matrices

Recall a matrix A can be decomposed using SVD

$$A = USV^{\mathrm{T}} \ . \tag{11}$$

By setting $m = n = 100$, the matrices U and V were construced randomly

$$U = \texttt{orth}(\texttt{randn}(100)) \tag{12}$$

$$V = \texttt{orth}(\texttt{randn}(100)) \ . \tag{13}$$

Simulating matrices

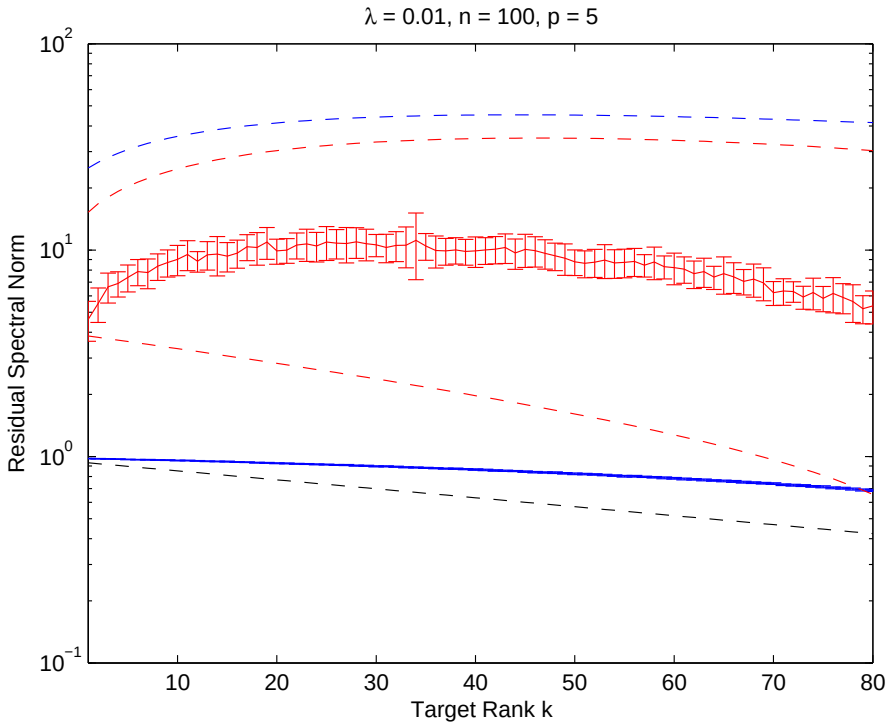
The singular values were predetermined to either to be exponentially decaying

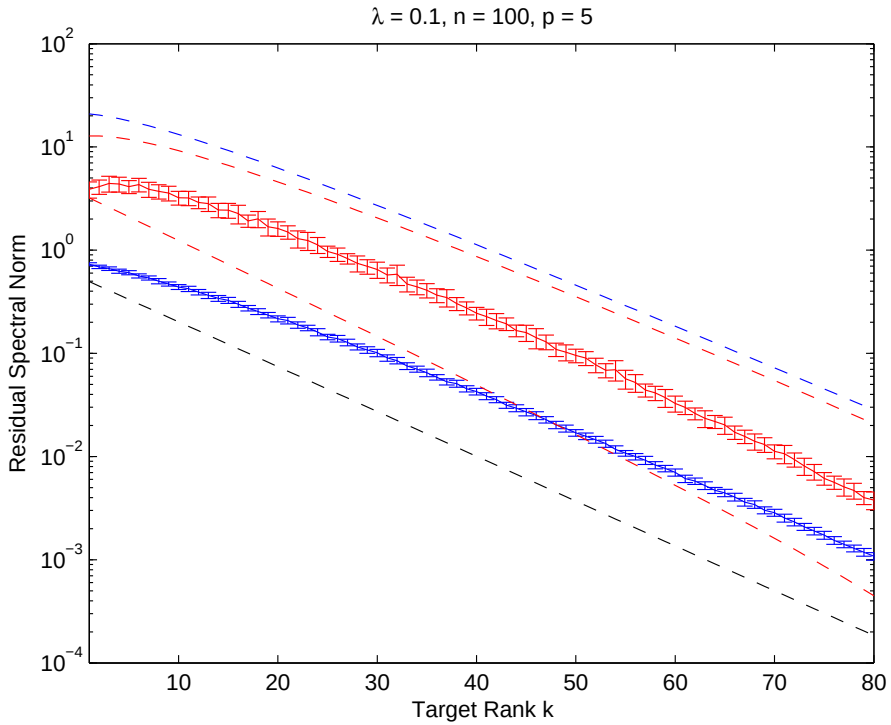
$$S = \begin{pmatrix} \mathrm{e}^{-\lambda \cdot 1} & 0 & \cdots & 0 \\ 0 & \mathrm{e}^{-\lambda \cdot 2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathrm{e}^{-\lambda \cdot 100} \end{pmatrix} \tag{14}$$

or has a sudden drop in value

$$S = \begin{pmatrix} I_{50} & 0 \\ 0 & \beta I_{50} \end{pmatrix} \tag{15}$$

Exponentially decaying singular values



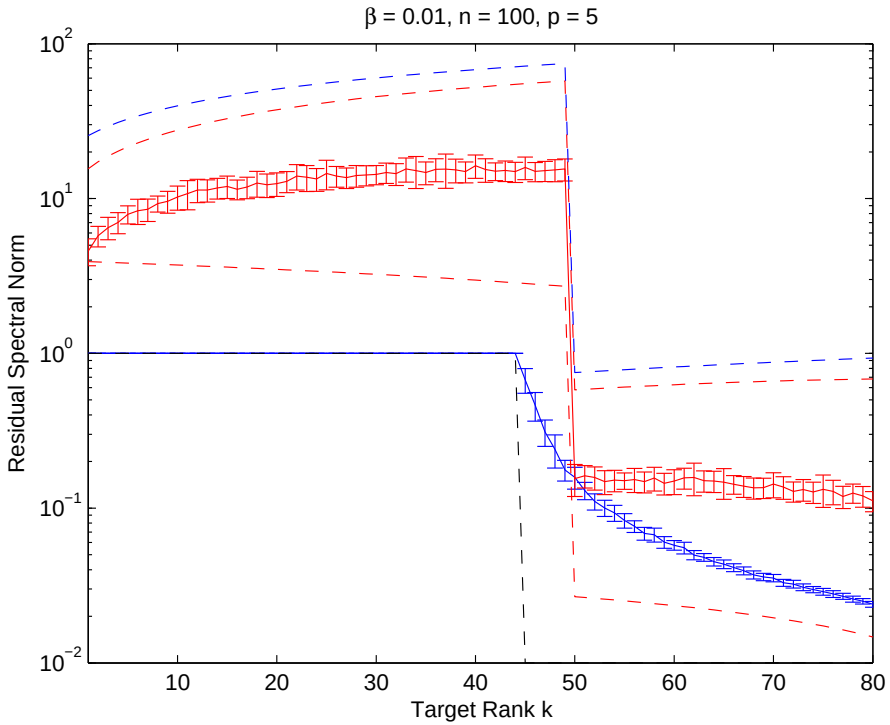


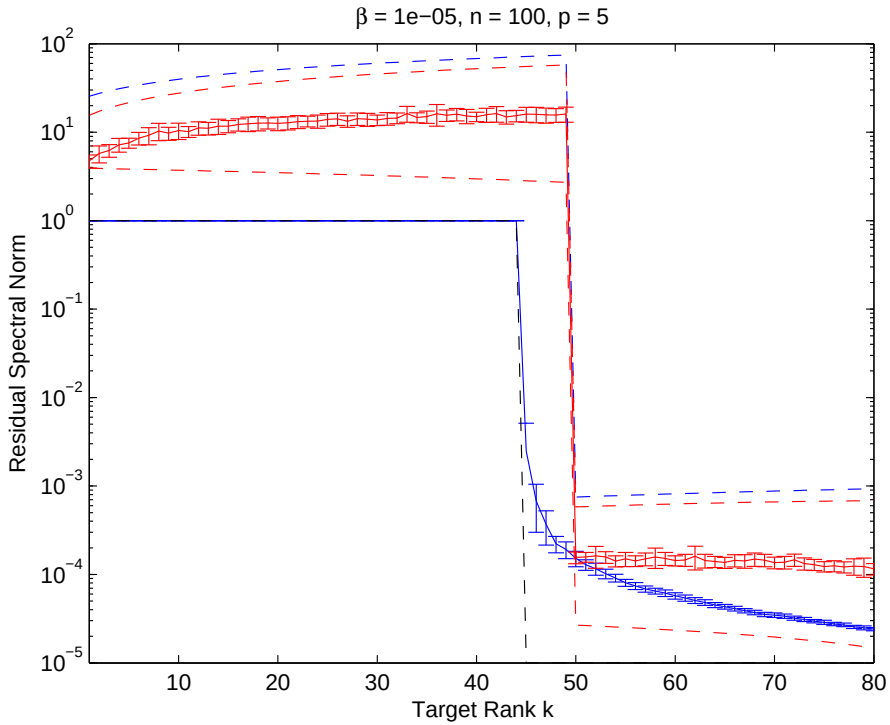
Blue line: Randomised algorithm on a matrix

Red line: Randomised algorithm on the worst case matrix

Black line: Truncated SVD

Sudden drop singular values





Blue line: Randomised algorithm on a matrix

Red line: Randomised algorithm on the worst case matrix

Black line: Truncated SVD

Thank you!