An Algorithm to Compute the Matrix Logarithm and its Fréchet Derivative for use in Condition Number Estimation

Awad Al-Mohy[1], Nicholas J. Higham[2] and Samuel Relton[2]

1. King Khalid University, SA
2. University of Manchester, UK

June 21, 2012

Overview

- The Principal Logarithm, Applications and Condition Numbers
- Inverse Scaling and Squaring Algorithms
- Backward Error Analysis
- Computation of Fréchet Derivatives for the Logarithm
- Using Real Arithmetic
- Numerical Experiments
- Conclusions

The Principal Matrix Logarithm

Definition (Principal Matrix Logarithm)

The principal matrix logarithm of $A \in \mathbb{C}^{n\times n}$ is a matrix $X \in \mathbb{C}^{n\times n}$ satisfying $e^X = A$ with eigenvalues in the strip $(-\pi, \pi)$ of the imaginary axis.

When $\rho(Y) < 1$ we have the series representation

$$\log(I + Y) = Y - \frac{Y^2}{2} + \frac{Y^3}{3} - \frac{Y^4}{4} + \cdots$$

from which we can obtain that $\log(A) \in \mathbb{R}^{n\times n}$ when $A \in \mathbb{R}^{n\times n}$.

Applications

Some recent applications of the matrix logarithm include:

- Transition rates between credit ratings
- Smooth computer animations
- Reduced-order models for real-time engineering
- Machine learning algorithms
- Manifold optimization techniques

Condition Numbers

Let $f(A) \in \mathbb{C}^{n \times n}$ be a matrix function.

Definition (Fréchet derivative)

The Fréchet derivative of f at A in direction $E \in \mathbb{C}^{n \times n}$ is the unique linear function $L(A, \cdot)$ such that

$$f(A + E) - f(A) - L(A, E) = o(\|E\|).$$

Condition Numbers

Let $f(A) \in \mathbb{C}^{n \times n}$ be a matrix function.

Definition (Fréchet derivative)

The Fréchet derivative of f at A in direction $E \in \mathbb{C}^{n \times n}$ is the unique linear function $L(A, \cdot)$ such that

$$f(A + E) - f(A) - L(A, E) = o(\|E\|).$$

Definition (Condition number)

The relative condition number of f at A is

$$\text{cond}(f, A) = \frac{\|L(A)\|\|A\|}{\|f(A)\|},$$

where

$$\|L(A)\| = \max_{E \neq 0} \frac{\|L(A, E)\|}{\|E\|}.$$

Condition Numbers

The *Kronecker form* $K(A) \in \mathbb{C}^{n^2 \times n^2}$ of the Fréchet derivative satisfies

$$\operatorname{vec}(L(A, E)) = K(A) \operatorname{vec}(E),$$

for any E.

Condition Numbers

The *Kronecker form* $K(A) \in \mathbb{C}^{n^2 \times n^2}$ of the Fréchet derivative satisfies

$$\operatorname{vec}(L(A, E)) = K(A) \operatorname{vec}(E),$$

for any E.

We will estimate $\|K(A)\|_1 \approx \|L(A)\|_1$ using the `normest1` algorithm in MATLAB, written by Higham and Tisseur. This method:

Condition Numbers

The *Kronecker form* $K(A) \in \mathbb{C}^{n^2 \times n^2}$ of the Fréchet derivative satisfies

$$\operatorname{vec}(L(A, E)) = K(A) \operatorname{vec}(E),$$

for any E.

We will estimate $\|K(A)\|_1 \approx \|L(A)\|_1$ using the `normest1` algorithm in MATLAB, written by Higham and Tisseur. This method:

- uses the 1-norm power method.
- involves around 8 Fréchet derivative evaluations.
- is accurate up to a factor n.

Inverse Scaling and Squaring

Kenney and Laub's ISS method exploits $\log(A) = 2^s \log(A^{1/2^s})$.

- Take s square roots of A.
- Use a diagonal Padé approximant $r_m(X)$ to $\log(I + X)$.
- Overall $\log(A) \approx 2^s r_m(A^{1/2^s} - I)$.

Inverse Scaling and Squaring

Kenney and Laub's ISS method exploits $\log(A) = 2^s \log(A^{1/2^s})$.

- Take s square roots of A.
- Use a diagonal Padé approximant $r_m(X)$ to $\log(I + X)$.
- Overall $\log(A) \approx 2^s r_m(A^{1/2^s} - I)$.

Repeatedly applying the chain rule to $\log(A) = 2\log(A^{1/2})$ we get

$$L_{\log}(A, E) = 2^s L_{\log}(A^{1/2^s}, E_s),$$

where E_s satisfies

$$A^{1/2^k} E_k + E_k A^{1/2^k} = E_{k-1}, \quad E_0 = E, \quad k = 1 : s.$$

The Basic Algorithm

1 $E_0 = E$
2 for $k = 1{:}\, s$
3 Compute $A^{1/2^k}$
4 Solve the Sylvester equation $A^{1/2^k} E_k + E_k A^{1/2^k} = E_{k-1}$
5 end
6 $\log(A) \approx 2^s r_m(A^{1/2^s} - I)$
7 $L_{\log}(A, E) \approx 2^s L_{r_m}(A^{1/2^s} - I, E_s)$

The Basic Algorithm

1 $E_0 = E$
2 for $k = 1{:}\, s$
3 Compute $A^{1/2^k}$
4 Solve the Sylvester equation $A^{1/2^k} E_k + E_k A^{1/2^k} = E_{k-1}$
5 end
6 $\log(A) \approx 2^s r_m(A^{1/2^s} - I)$
7 $L_{\log}(A, E) \approx 2^s L_{r_m}(A^{1/2^s} - I, E_s)$

- Notice approximation $L_{\log} \approx L_{r_m}$ in line 7
- Calculate multiple Fréchet derivatives to get condition number
- *Reuse* computation for $\log(A)$ in $L_{\log}(A, E)$
- Use *Schur decomp.* for increased accuracy and performance

Backward Error Analysis

Taking $X = A^{1/2^s}$ Al-Mohy and Higham define

$$h_{2m+1}(X) = e^{r_m(X)} - X - I = \sum_{k=2m+1}^{\infty} c_k X^k,$$

giving the *error in the Padé approximant*.

Backward Error Analysis

Taking $X = A^{1/2^s}$ Al-Mohy and Higham define

$$h_{2m+1}(X) = e^{r_m(X)} - X - I = \sum_{k=2m+1}^{\infty} c_k X^k,$$

giving the *error in the Padé approximant*.

Also they let

$$\alpha_p(X) = \max(\|X^p\|^{1/p}, \|X^{p+1}\|^{1/(p+1)}),$$

where $p \in \mathbb{N}$ satisfies $p(p-1) \leq 2m+1$, noting that $\alpha_p(X) \leq \|X\|$. When X is non-normal we can have $\alpha_p(X) \ll \|X\|$.

Backward Error Analysis

Al-Mohy and Higham derive the following:

Theorem (Backward Error for log)

Let $\rho(r_m(X)) < \pi$ and p satisfy the above. Then $r_m(X) = \log(I + X + \Delta X)$ and

$$\frac{\|\Delta X\|}{\|X\|} \leq \sum_{k=2m+1}^{\infty} |c_k| \alpha_p(X)^{k-1},$$

where $\Delta X = h_{2m+1}(X)$.

Backward Error Analysis

Al-Mohy and Higham derive the following:

Theorem (Backward Error for log)

Let $\rho(r_m(X)) < \pi$ *and* p *satisfy the above. Then* $r_m(X) = \log(I + X + \Delta X)$ *and*

$$\frac{\|\Delta X\|}{\|X\|} \leq \sum_{k=2m+1}^{\infty} |c_k| \alpha_p(X)^{k-1},$$

where $\Delta X = h_{2m+1}(X)$.

Theorem (Backward Error for $L_{\log}$)

Under the same conditions for any $E \in \mathbb{C}^{n \times n}$ $L_{r_m}(X, E) = L_{\log}(I + X + \Delta X, E + \Delta E)$ *where* ΔX *is as before and* $\Delta E = L_{h_{2m+1}}(X, E)$.

Bounding ΔE

Bounds θ_m such that $\alpha_p(X) \leq \theta_m$ imply $\|\Delta X\|/\|X\| \leq u$ are given by Al-Mohy and Higham. *Can we do the same for the Fréchet derivatives?*

Bounding ΔE

Bounds θ_m such that $\alpha_p(X) \leq \theta_m$ imply $\|\Delta X\|/\|X\| \leq u$ are given by Al-Mohy and Higham. *Can we do the same for the Fréchet derivatives?*

Expressing ΔE as a power series we know that

$$\Delta E = \sum_{k=2m+1}^{\infty} c_k \sum_{j=1}^{k} X^{j-1} E X^{k-j},$$

Bounding ΔE

Bounds θ_m such that $\alpha_p(X) \leq \theta_m$ imply $\|\Delta X\|/\|X\| \leq u$ are given by Al-Mohy and Higham. *Can we do the same for the Fréchet derivatives?*

Expressing ΔE as a power series we know that

$$\Delta E = \sum_{k=2m+1}^{\infty} c_k \sum_{j=1}^{k} X^{j-1} E X^{k-j},$$

and taking norms:

$$\frac{\|\Delta E\|}{\|E\|} \leq \sum_{k=2m+1}^{\infty} k|c_k|\|X\|^{k-1}.$$

Bounding ΔE

Using Al-Mohy and Higham's bounds for $\|\Delta X\|/\|X\| \leq u$ to bound the Fréchet derivative, *how large* can $\|\Delta E\|/\|E\|$ be?

Bounding ΔE

Using Al-Mohy and Higham's bounds for $\|\Delta X\|/\|X\| \leq u$ to bound the Fréchet derivative, *how large* can $\|\Delta E\|/\|E\|$ be?

m	1	6	7
μ_m	4	15.444	18.545

- For a Padé approximation degree m, $\|\Delta E\|/\|E\| \leq \mu_m u$.
- The values of μ_m increase with m.
- Algorithm normally chooses $m = 6$ or 7.

Computing $L_{r_m}(X, E)$

When computing the matrix logarithm we use the *partial fraction* representation of the Padé approximant.

Differentiating we obtain

$$L_{r_m}(X, E_s) = \sum_{j=1}^{m} L_{r_m^j}(X, E_s),$$

where

$$L_{r_m^j}(X, E_s) = \alpha_j^{(m)}(I + \beta_j^{(m)} X)^{-1} E_s (I + \beta_j^{(m)} X)^{-1}.$$

Computing $L_{r_m}(X, E)$

When computing the matrix logarithm we use the *partial fraction* representation of the Padé approximant.

Differentiating we obtain

$$L_{r_m}(X, E_s) = \sum_{j=1}^{m} L_{r_m^j}(X, E_s),$$

where

$$L_{r_m^j}(X, E_s) = \alpha_j^{(m)} (I + \beta_j^{(m)} X)^{-1} E_s (I + \beta_j^{(m)} X)^{-1}.$$

- *Cost:* $(8 + 2s + 2m)\, n^3$ flops for each Fréchet derivative.
- We can store and *reuse* the $A^{1/2^k}$, required to compute $\log(A)$, in computing $L_{\log}(A, E)$.
- Estimating the condition number with `normest1` requires around 8 Fréchet derivative evaluations.

Using Real Arithmetic

When $A, E \in \mathbb{R}^{n \times n}$ both $\log(A)$ and $L_{\log}(A, E)$ are real. Using the *real Schur decomp.* we have some 2×2 diagonal blocks of the form

$$B = \begin{bmatrix} a & b \\ c & a \end{bmatrix}, \quad bc < 0,$$

with eigenvalues $\lambda_{\pm} = a \pm i(-bc)^{1/2}$.

Using Real Arithmetic

When $A, E \in \mathbb{R}^{n\times n}$ both $\log(A)$ and $L_{\log}(A, E)$ are real. Using the *real Schur decomp.* we have some 2×2 diagonal blocks of the form

$$B = \begin{bmatrix} a & b \\ c & a \end{bmatrix}, \quad bc < 0,$$

with eigenvalues $\lambda_{\pm} = a \pm i(-bc)^{1/2}$.

We have obtained the *explicit representation*

$$\log(B) = \begin{bmatrix} \log(a^2 - bc)/2 & \theta b(-bc)^{-1/2} \\ \theta c(-bc)^{-1/2} & \log(a^2 - bc)/2 \end{bmatrix},$$

where $\theta = \arg(\lambda_+) \in (0, \pi)$, which avoids all *subtractive cancellation*.

Using Real Arithmetic

When $A, E \in \mathbb{R}^{n \times n}$ both $\log(A)$ and $L_{\log}(A, E)$ are real. Using the *real Schur decomp.* we have some 2×2 diagonal blocks of the form

$$B = \begin{bmatrix} a & b \\ c & a \end{bmatrix}, \quad bc < 0,$$

with eigenvalues $\lambda_{\pm} = a \pm i(-bc)^{1/2}$.

We have obtained the *explicit representation*

$$\log(B) = \begin{bmatrix} \log(a^2 - bc)/2 & \theta b(-bc)^{-1/2} \\ \theta c(-bc)^{-1/2} & \log(a^2 - bc)/2 \end{bmatrix},$$

where $\theta = \arg(\lambda_+) \in (0, \pi)$, which avoids all *subtractive cancellation*.

- *Increased speed* using real flops over complex flops of factor 2-3.
- *Decreased memory footprint* storing matrices.
- *Decreased rounding errors* due to small imaginary parts.

Other Methods to Compute $L_{\log}(A, E)$

Kenney and Laub introduce `kron_sylv` which solves a number of Sylvester equations then taking a Padé approximation to $\tanh(x)/x$.

We denote by `kron_sylv_mod` the same algorithm as above, replacing all calls to `logm` with `iss_schur_complex` by Al-Mohy and Higham.

Other Methods to Compute $L_{\log}(A, E)$

Kenney and Laub introduce `kron_sylv` which solves a number of Sylvester equations then taking a Padé approximation to $\tanh(x)/x$.

We denote by `kron_sylv_mod` the same algorithm as above, replacing all calls to `logm` with `iss_schur_complex` by Al-Mohy and Higham.

The `integral` method by Dieci et al. solves an integral formulation of $L_{\log}(A, E)$ using an adaptive Simpson scheme.

Other Methods to Compute $L_{\log}(A, E)$

Kenney and Laub introduce `kron_sylv` which solves a number of Sylvester equations then taking a Padé approximation to $\tanh(x)/x$.

We denote by `kron_sylv_mod` the same algorithm as above, replacing all calls to `logm` with `iss_schur_complex` by Al-Mohy and Higham.

The `integral` method by Dieci et al. solves an integral formulation of $L_{\log}(A, E)$ using an adaptive Simpson scheme.

Finally `dbl_size` calculates

$$\log\left(\begin{bmatrix} A & E \\ 0 & A \end{bmatrix}\right) = \begin{bmatrix} \log(A) & L_{\log}(A, E) \\ 0 & \log(A) \end{bmatrix}.$$

Other Methods to Compute $L_{\log}(A, E)$

Kenney and Laub introduce `kron_sylv` which solves a number of Sylvester equations then taking a Padé approximation to $\tanh(x)/x$.

We denote by `kron_sylv_mod` the same algorithm as above, replacing all calls to `logm` with `iss_schur_complex` by Al-Mohy and Higham.

The `integral` method by Dieci et al. solves an integral formulation of $L_{\log}(A, E)$ using an adaptive Simpson scheme.

Finally `dbl_size` calculates

$$\log\left(\begin{bmatrix} A & E \\ 0 & A \end{bmatrix}\right) = \begin{bmatrix} \log(A) & L_{\log}(A, E) \\ 0 & \log(A) \end{bmatrix}.$$

- All these methods are *more expensive* than `iss_schur`.

Numerical Experiments

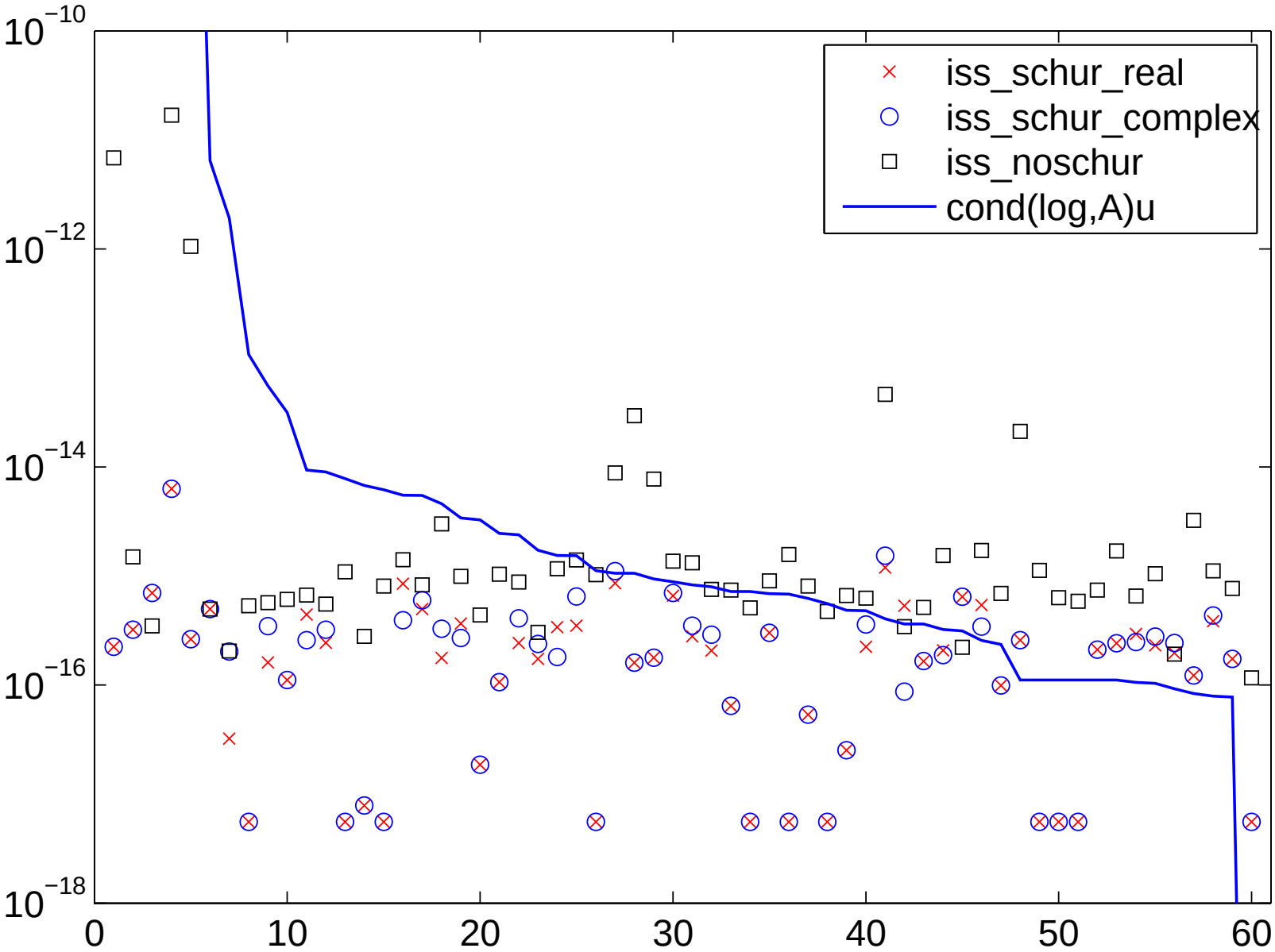
We perform a number of experiments against existing algorithms:

- Computing the matrix logarithm.
- Computing the Fréchet derivatives.
- Estimating the condition number.

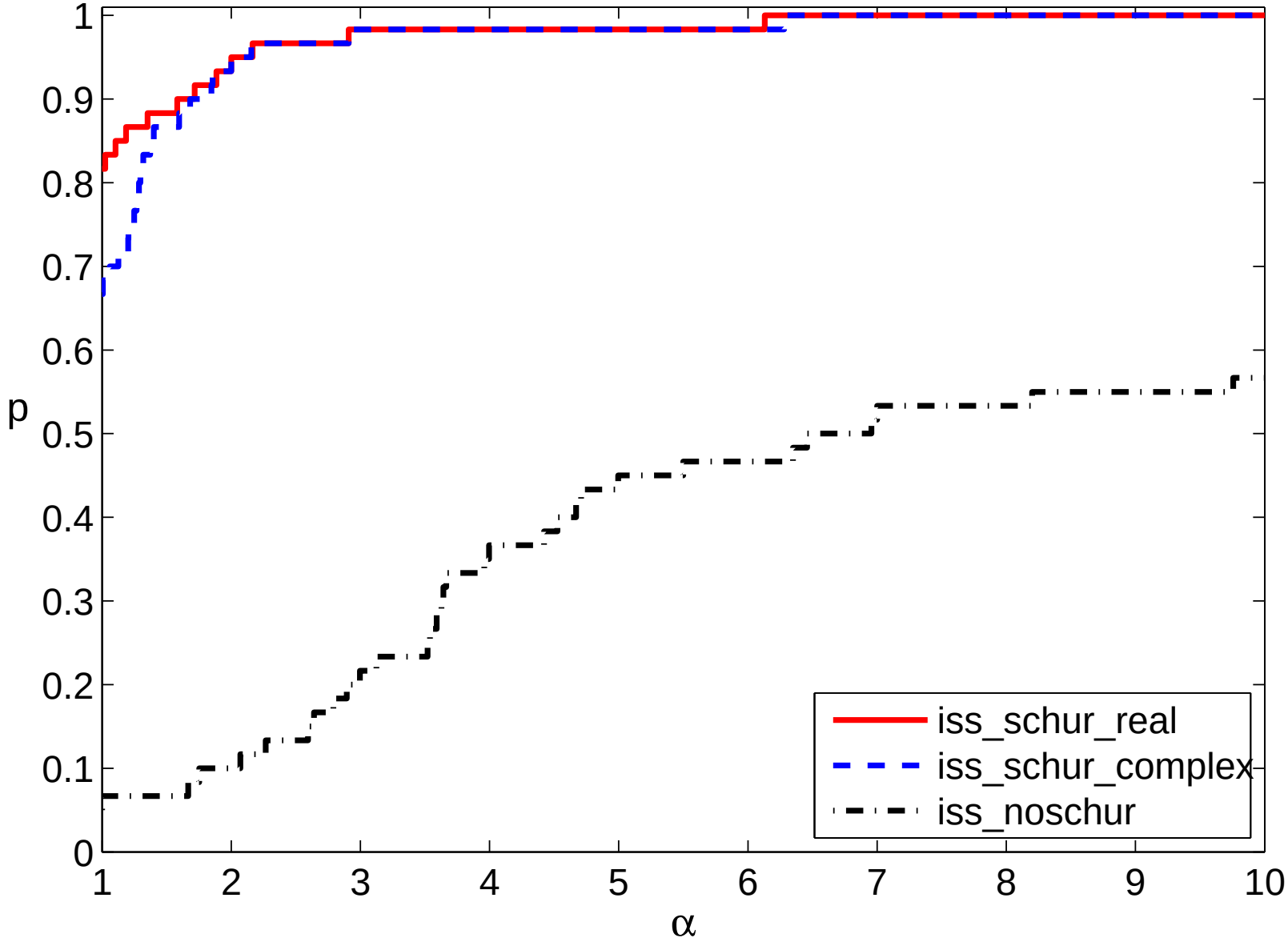
The test matrices are taken from the literature and the Matrix Computation Toolbox.

- Mostly 10×10 matrices (60/66 real matrices).
- Some very ill conditioned problems.
- Schur decomposition used to avoid introducing error. Similar results for full matrices.

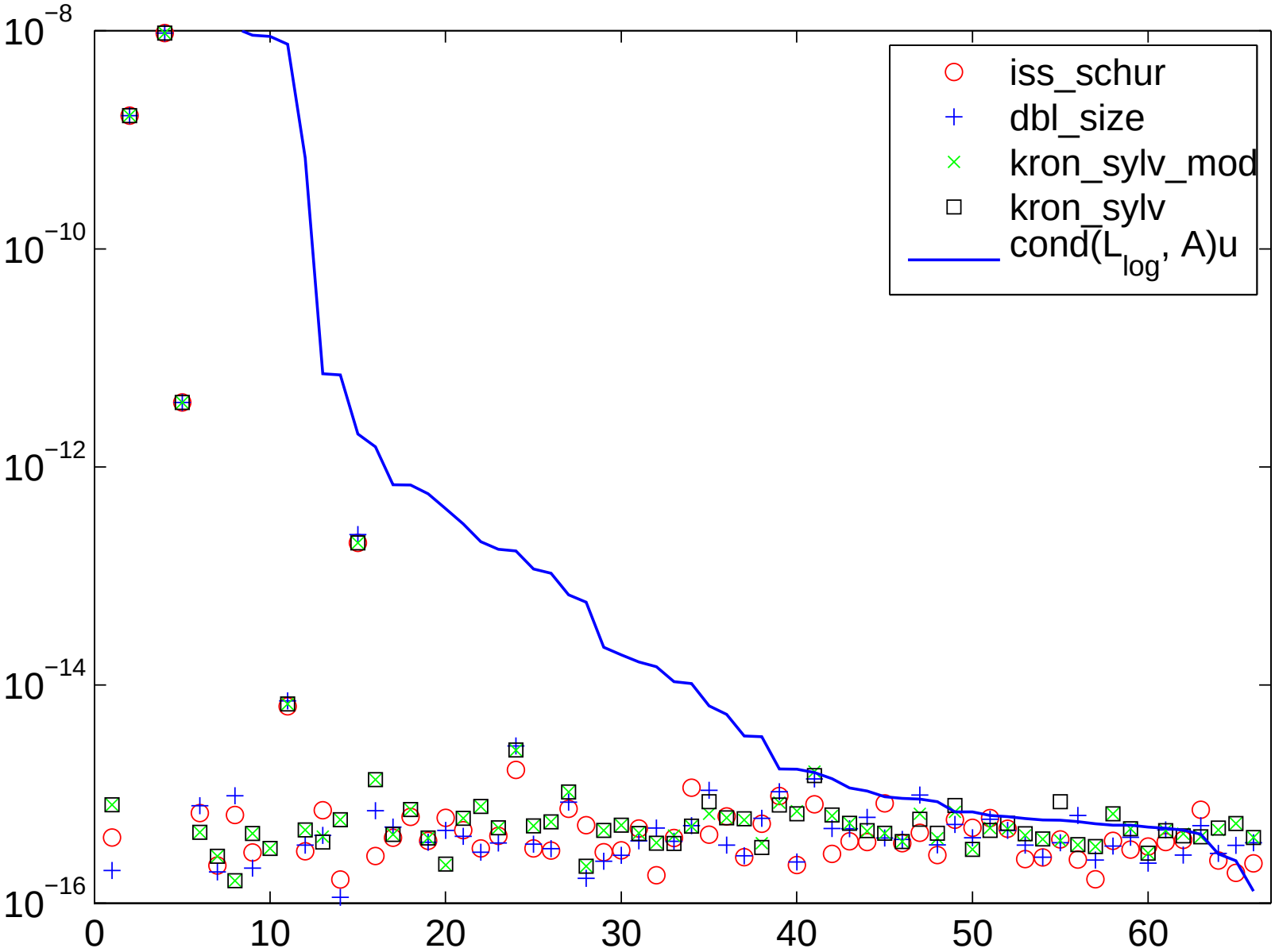
Computing the Logarithm - Relative Errors



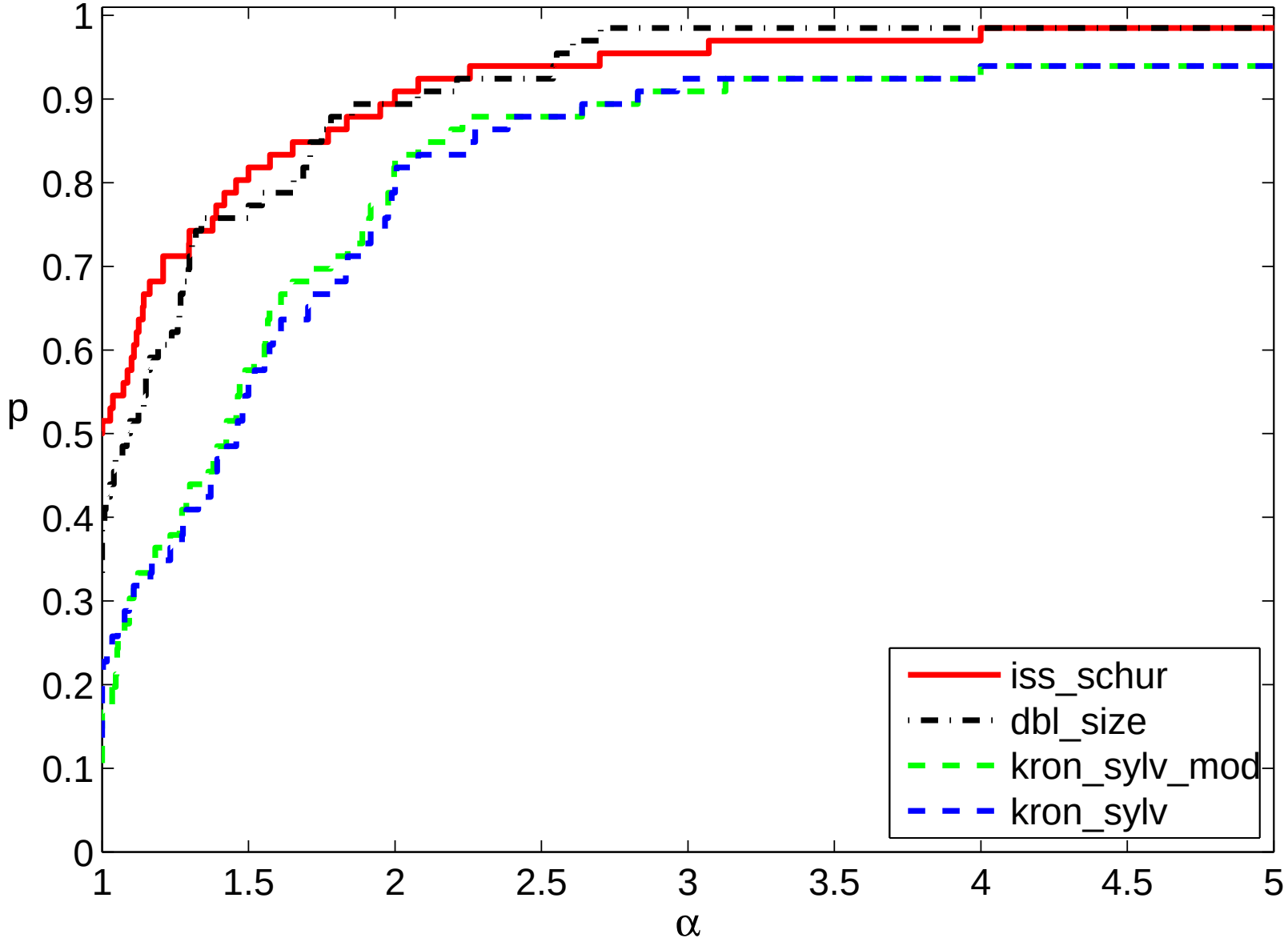
Computing the Logarithm - Performance Profile



Computing $L_{\log}(A, E)$ - Relative Errors

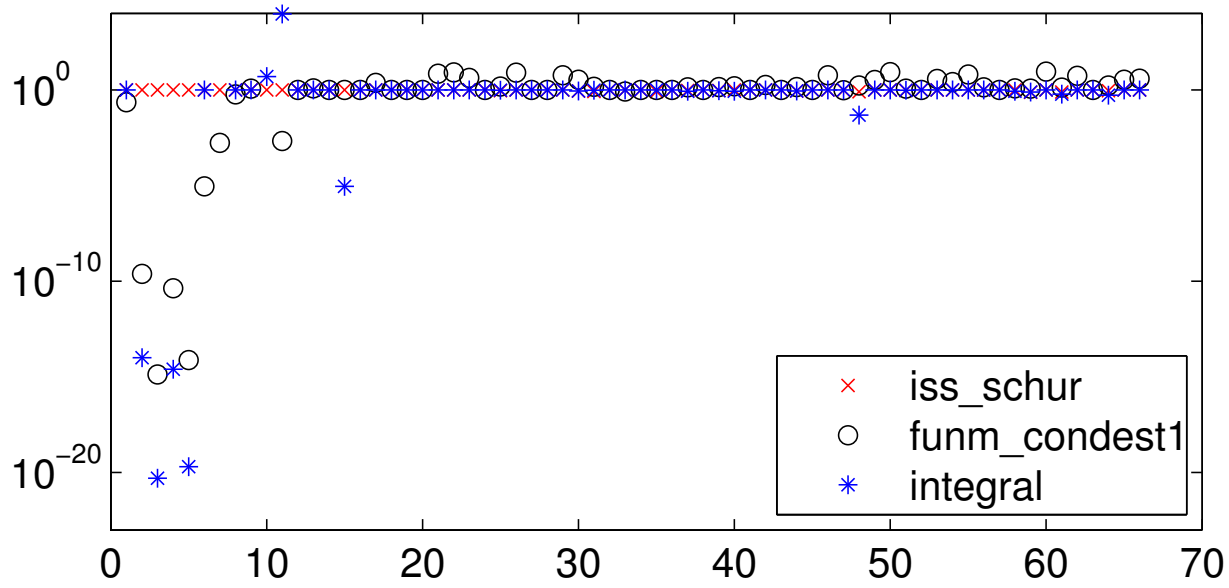


Computing $L_{\log}(A, E)$ - Performance Profile



Computing the condition number- Ratios

These plots show the ratios $\kappa_{\text{est}}/\kappa_{\text{exact}} \leq 1$.

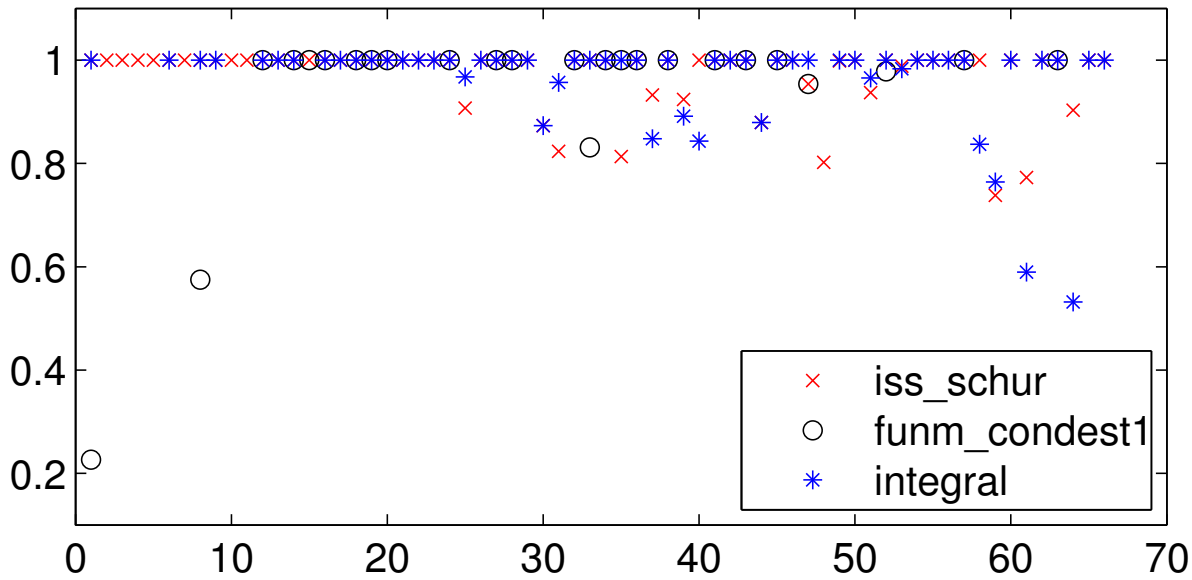


- On the badly conditioned problems both `integral` and `funm_condest1` greatly *underestimate* the condition number.
- Our method remains stable throughout.

Computing the condition number- Ratios

Zoomed version of the previous figure.

- `iss_schur` is more *stable* than alternatives.
- Other methods are *prohibitively expensive* in comparison.



Conclusions

- New algorithm, using *real arithmetic* when A is real, result in *faster* and more *accurate* computation of $\log(A)$ using *less memory*.
- New algorithms for $L_{\log}(A, E)$ are *significantly less expensive* than existing algorithms and are *more accurate* in practice.
- By *reusing previous computation* we allow efficient evaluation of multiple Fréchet derivatives allowing accurate condition number estimation.

References

- Al-Mohy, Higham and Relton - *Computing the Fréchet derivative of the Matrix Logarithm and Estimating the Condition Number*, MIMS EPrint 2012.72
- Al-Mohy and Higham - *Computing the Fréchet derivative of the matrix exponential with an application to condition number estimation*, SIMAX 2009
- Al-Mohy and Higham - *Improved inverse scaling and squaring algorithms for the matrix logarithm*, To appear in SISC 2012
- Dieci, Morini and Papini - *Computational techniques for real logarithms of matrices*, SIMAX 1996
- Higham - *Evaluating Padé approximants of the matrix logarithm*, SIMAX 2001
- Higham - *Functions of Matrices: Theory and Computation*, 2008
- Kenney and Laub - *Condition estimates for matrix functions*, SIMAX 1989
- Kenney and Laub - *A Schur-Fréchet algorithm for computing the logarithm and exponential of a matrix*, SIMAX 1998