MANCHESTER 1824

The University of Manchester

M\crNA
Manchester Numerical Analysis

A Multiprecision Derivative-Free Schur–Parlett Algorithm for Computing Matrix Functions

Nicholas J. Higham*, Xiaobo Liu*

*Department of Mathematics, The University of Manchester, UK

SIAM Conference on Applied Linear Algebra, May 20, 2021

Outline

- Background
  - The Schur–Parlett algorithm
  - Davies's randomized approximate diagonalization (AD)
    - Unreliability
    - Experimental results
- Computing a function of a triangular matrix
  - AD with triangular perturbation
  - Experimental results
- The multiprecision Schur–Parlett algorithm
  - Experimental results

Definition of Matrix Function via Jordan Canonical Form

- For $A \in \mathbb{C}^{n\times n}$ there exists invertible $Z \in \mathbb{C}^{n\times n}$ such that

$$Z^{-1}AZ = J = \operatorname{diag}(J_1, \ldots, J_p),\ J_k = \begin{bmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{bmatrix} \in \mathbb{C}^{m_k \times m_k},$$

where $m_1 + m_2 + \cdots + m_p = n$.

- Let $A \in \mathbb{C}^{n\times n}$ and let f be defined on the spectrum of A. Then $f(A) = Z \operatorname{diag}(f(J_1), \ldots, f(J_p))Z^{-1}$, where

$$f(J_k) := \begin{bmatrix} f(\lambda_k) & f'(\lambda_k) & \cdots & \frac{f^{(m_k-1)}(\lambda_k)}{(m_k-1)!} \\ & f(\lambda_k) & \ddots & \vdots \\ & & \ddots & f'(\lambda_k) \\ & & & f(\lambda_k) \end{bmatrix}.$$

The Schur–Parlett Algorithm (Davies & Higham, 2003)

- Compute the Schur decomposition $A = QTQ^*$.
- Partition and reorder $T = (T_{ij})$ to blocked upper triangular $\widetilde{T} = U^* TU$, with a blocking parameter $\delta > 0$, such that
  - E'vals from different diagonal blocks are well separated ($|\lambda - \mu| > \delta$ for λ and μ from distinct diagonal blocks).
  - E'vals in the same diagonal block are well clustered ($|\lambda_1 - \lambda_2| \leq \delta$ for e'vals in the same block).
- Evaluate $F_{ii} = f(\widetilde{T}_{ii})$ by truncating the Taylor series $f(\widetilde{T}_{ii}) = \sum_{k=0}^{\infty} \frac{f^{(k)}(\sigma)}{k!}(\widetilde{T}_{ii} - \sigma I)^k, \sigma = \mathrm{trace}(\widetilde{T}_{ii})/n$.
- Solve the triangular Sylvester equation for F_{ij}

$$T_{ii}F_{ij} - F_{ij}T_{jj} = F_{ii}T_{ij} - T_{ij}F_{jj} + \sum_{k=i+1}^{j-1}(F_{ik}T_{kj} - T_{ik}F_{kj}), \quad i < j.$$

- $f(A) = Qf(T)Q^* = QUFU^*Q^*$

Randomized Approximate Diagonalization (Davies, 2007)

• Diagonalization for computing f of a diagonalizable $A \in \mathbb{C}^{n\times n}$: compute $A = V_0 D_0 {V_0}^{-1}$ and

$$f(A) = V_0 f(D_0) V_0^{-1}.$$

• For (highly) nonnormal $A \in \mathbb{C}^{n\times n}$, diagonalization after a **(full) random** perturbation E: compute $A + E = \widetilde{A} = VDV^{-1}$ and

$$f(A) \approx f(\widetilde{A}) = Vf(D)V^{-1}. \tag{1}$$

This succeeds with probability 1.

How to choose the E?

How accurate is the approximation in (1)?

Davies's Conjecture

Davies measured the quality of the approximate diagonalization by

$$\sigma(A, V, E, \epsilon) = \kappa_2(V)\epsilon + \|E\|_2,$$

and conjectured that

$$\sigma(A, \epsilon) = \inf_{E,V} \sigma(A, V, E, \epsilon) \leq c_n\sqrt{\epsilon}$$

for some c_n, where $\|A\|_2 \leq 1$ is assumed.

- **The conjecture has been proved with $\|E\|_2 \leq \sqrt{\epsilon}$** (Banks & Kulkarni & Mukherjee & Srivastava, 2020).
- Taking E with $\|E\|_2 \approx \sqrt{\epsilon}$ **often** gives an error of $O(\sqrt{\epsilon})$ for $\|A\|_2 \leq 1$.

For small E, $\|f(A+E) - f(A)\| \lesssim \|L_f(A, E)\| \leq \|L_f(A)\| \, \|E\|$, where $\|L_f(A)\| = \max\{ \|L_f(A, E)\| : \|E\| = 1 \}$.

- **$\|L_f(A)\|_2$ can greatly exceed 1!**
- Flawed definition of $\sigma(A, V, E, \epsilon) = \kappa_2(V)\epsilon + \|E\|_2$ in computation.

Approximate Diagonalization with Triangular Perturbation

In the (standard) Schur–Parlett:

$F_{ii} = f(T_{ii}) = \sum_{k=0}^{\infty} \frac{f^{(k)}(\sigma)}{k!}(T_{ii} - \sigma I)^k, \sigma = \mathrm{trace}(T_{ii})/n.$

In the derivative-free Schur–Parlett:

Diagonalize $T + E$ for a random **triangular perturbation** E.

Advantages:

- Diagonalization succeeds with probability 1.
- E of order $u\,\|T\|$ does no harm (full E does, in b'ward sense), where u: unit roundoff of the working precision.
- Only need to compute the eigensystem of a triangular matrix $\widetilde{T} = T + E$ of size $m \times m$.

How accurate?

Approximate Diagonalization with Triangular Perturbation

The error in the computed approximation $\widehat{F} \approx F = f(\widetilde{T})$ satisfies (Higham, 2008)

$$\frac{\|F - \widehat{F}\|_1}{\|F\|_1} \lesssim \kappa_1(V) \frac{\|f(D)\|_1}{\|f(\widetilde{T})\|_1} u_h \leq \kappa_1(V) u_h,$$

where u_h is the (unit roundoff of) the precision of diagonalization.

- From $\|F - \widehat{F}\|_1 / \|F\|_1 \lesssim u$, for large $\kappa_1(V)$ we need $u_h < u$.

- Access to arbitrary precision: Advanpix Multiprecision Computing Toolbox for MATLAB

How to choose u_h — estimating $\kappa(V)$ based only on $\widetilde{T}$?

Determining the Precision

We have (Demmel, 1983)

$$\kappa_2(V) \leq m \cdot \max_i \|P_i\|_2,$$

and

$$\|P_1\|_1 \leq \max\big(1, \|\widetilde{t}_{12}\|_\infty \|(\widetilde{t}_{11} I - \widetilde{T}_{22})^{-1}\|_1\big),$$

where P_i is the spectral projector corresponding to the e'val λ_i, and

$$\widetilde{T} = \begin{bmatrix} \widetilde{t}_{11} & \widetilde{t}_{12}^* \\ 0 & \widetilde{T}_{22} \end{bmatrix}.$$

- **Approximate** $\|(\widetilde{t}_{11} I - \widetilde{T}_{22})^{-1}\|_1$

Determining the Precision

For any $m \times m$ upper triangular matrix U we have (Higham, 2002)

$$\|U^{-1}\|_1 \leq \frac{1}{\alpha}\left(\frac{\beta}{\alpha}+1\right)^{m-1}, \quad \alpha = \min_i |u_{ii}|, \quad \beta = \max_{i<j} |u_{ij}|. \tag{2}$$

Brief idea:

- Group $\widetilde{t}_{ii}$ with parameter $\delta_1 < \delta$, largest block has size k.
- approximate $\|(\widetilde{t}_{11}I - \widetilde{T}_{22})^{-1}\|_1$ by $\|(\widetilde{t}_{11}I - \widetilde{T}_{22}(1\colon k-1, 1\colon k-1))^{-1}\|_1$, and bound it by (2).

The approximation gives the requirement

$$u_h \lesssim \frac{c_m u^2}{\max_{i<j} |\widetilde{t}_{ij}| \left(\frac{\max_{i<j} |\widetilde{t}_{ij}|}{c_m u} + 1\right)^{k-2}}, \quad k \geq 2.$$

- **Parameters δ_1 and c_m**

Multiprecision Algorithm for Function of a Triangular Matrix

Algorithm 1: Given triangular matrix $T \in \mathbb{C}^{m\times m}$, this algorithm computes $F = f(T)$.

1		If $m = 1$ or $m = 2$ and $t_{11} \neq t_{22}$, use explicit formula for $f(T)$, quit.
2		Form a diagonal or upper triangular standard Gaussian N.
3		$E = u(\max_{i,j} \lvert t_{ij}\rvert / \lVert N\rVert_F)N$
4	$\mathbf{u^2}$	$\widetilde{T} = T + E$
5	$\mathbf{u^2}$	$D = \mathrm{diag}(\widetilde{T})$
6	$\mathbf{u^2}$	Evaluate u_h.
7	$\mathbf{u_h}$	if $\mathbf{u_h} < \mathbf{u^2}$, convert $\widetilde{T}$ and D to precision $\mathbf{u_h}$.
8		for $i = 1 : m$
9	$\mathbf{u_h}$	Set $(v_i)_i = 1$ and $(v_i)_k = 0$ for $k > i$ and solve $(\widetilde{T} - \widetilde{t}_{ii} I)v_i = 0$ for the first $i - 1$ components of v_i.
10		end
11	$\mathbf{u_h}$	Form $F = Vf(D)V^{-1}$, where $V = [v_1, \ldots, v_m]$.
12		Round F to precision u.

Experimental Results

Equivalent number of **decimal digits** for $\mathbf{u_h}$ used by Algorithm 1 in the computation. 32 digits corresponds to $\mathbf{u_h} = \mathbf{u^2}$.

	$m = 35$	$m = 75$
T_1 = `gallery('kahan',m)`	32	623
T_2 = `schur(gallery('smoke',m),'complex')`	32	32
T_3 = `schur(randn(m),'complex')`	32	32
T_4 = `schur(rand(m),'complex')`	32	32
T_5 = `triu(randn(m))`	34	68
T_6 = `triu(rand(m))`	51	68
T_7 = `gallery('jordbloc',m,0.5)`	599	1296

- T_1 has distinct eigenvalues on $(0, 1]$.

- **In general m is not expected to be large.**

Experimental Results

Maximal errors for Algorithm 1 with a **diagonal** E and the method of approximate diagonalization with **full perturbation**. $f = \text{sqrt}$.

$m = 35$	`Alg_diag`	`Alg_full`	$\kappa_{\text{sqrt}}(A)u$
T_1	2.7e-16	3.1e-13	5.4e-11
T_2	5.9e-16	3.9e-13	5.6e-11
T_3	1.4e-16	1.4e-16	3.3e-14
T_4	1.4e-15	1.3e-15	4.9e-14
T_5	1.5e-15	**9.7e-9**	3.8e-10
T_6	1.0e-15	6.7e-12	5.6e-9
T_7	4.1e-16	**8.5e-8**	3.9e-12
$m = 75$	`Alg_diag`	`Alg_full`	$\kappa_{\text{sqrt}}(A)u$
T_1	2.1e-15	**8.2e-7**	3.2e-11
T_2	5.5e-16	**1.0e-7**	4.5e-12
T_3	1.8e-16	1.3e-16	1.5e-13
T_4	1.4e-15	1.7e-15	1.5e-13
T_5	2.5e-14	**1.0**	4.3e-22
T_6	1.9e-15	**1.0**	2.7e-14
T_7	3.4e-16	**1.0**	6.6e-18

Algorithm 2: Given $A \in \mathbb{C}^{n\times n}$ this algorithm computes $F = f(A)$.

```
Compute A = QTQ*.
If T is diagonal, F = Qf(T)Q*, quit.
Reorder T with δ > 0 to a block m × m upper triangular T̃ = U*TU.
for i = 1 : m
    Use Algorithm 1 to evaluate F_ii = f(T̃_ii).
    for j = i − 1 : −1 : 1
        Solve the Sylvester equation for F_ij.
    end
end
F = QUFU*Q*
```

Costs: $28n^3$ in precision u plus $2/3\sum_{i=1}^{s} m_i^3$ in precision $\mathbf{u_h}$.

- **Precision independent framework**

The Blocking Parameter $\delta > 0$

In the (standard) Schur–Parlett $\delta = 0.1$.

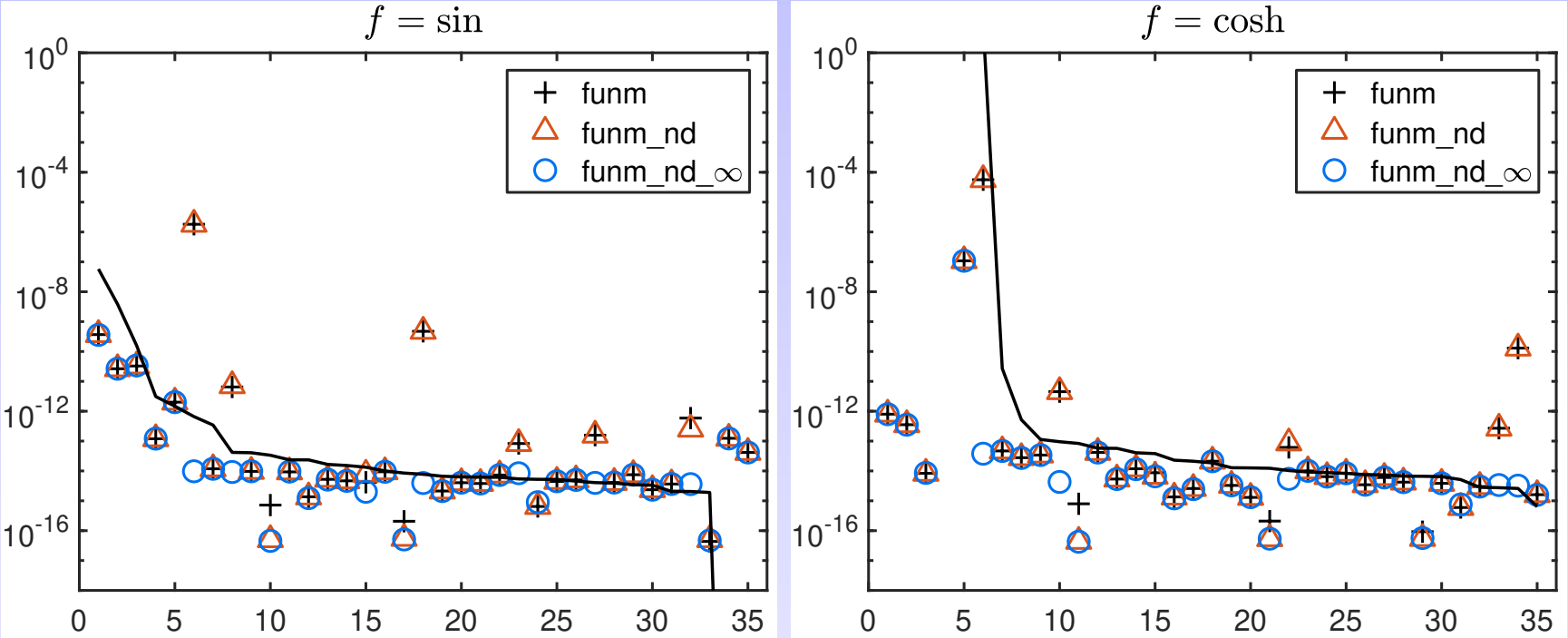
- This optimal choice of δ is **problem-dependent**.
- Too large a δ cause problems in the Taylor series approximation.

In Algorithm 2:

- $\delta = 0.1$ by default.
- Larger $\delta \Rightarrow$ better-conditioned Sylvester equations in general.
- Larger $\delta \Rightarrow$ potentially high precision used on larger blocks!
- When $\delta = \infty$, compute $f(T)$ in higher precision & no need to solve Sylvester equation (**optimally accurate but expensive**).

• Algorithm 2 with $\delta = \infty$ costs $28n^3$ in precision u plus $2/3n^3$ in precision $\mathbf{u_h}$.

Experimental Results



Relative errors on 35 matrices of size 32×32 from the MATLAB gallery and the Matrix Computation Toolbox.

- `funm`, the built-in MATLAB function implementing the Schur–Parlett.
- `funm_nd`, Algorithm 2 with $\delta = 0.1$.
- `funm_nd_∞`, Algorithm 2 with $\delta = \infty$.

Experimental Results

Results over 10 runs. Size: the maximal block size. Digits: the maximal number of equivalent decimal digits used by `funm_nd`.

	Maximal relative error			Mean execution time (secs)				
$n = 40$	`funm`	`funm_nd`	`funm_nd_∞`	`funm`	`funm_nd`	`funm_nd_∞`	size (m)	digits
A_1	4.6e-15	4.6e-15	4.6e-15	2.1e-2	4.5e-2	1.4e-1	8	32
A_2	4.0e-15	4.0e-15	3.9e-15	2.2e-2	2.4e-2	1.4e-1	3	32
A_3	**1.5e-14**	7.1e-17	6.8e-17	1.8e-3	4.1e-2	4.1e-2	40	685
$n = 100$	`funm`	`funm_nd`	`funm_nd_∞`	`funm`	`funm_nd`	`funm_nd_∞`	size (m)	digits
A_1	6.7e-15	6.7e-15	6.7e-15	6.6e-2	1.6e-1	9.7e-1	13	32
A_2	6.3e-15	6.4e-15	6.3e-15	1.9e-1	1.9e-1	1.0	4	32
A_3	**1.0e-12**	5.8e-17	5.8e-17	2.7e-2	7.3e-1	7.4e-1	100	1734

Test matrices:

- $A_1 =$ `rand(n)`$/5$.
- $A_2 =$ `randn(n)`$/10$.
- $A_3 =$ `gallery('triw',n,-5)`: upper triangular with 1s on the diagonal and -5s off the diagonal.

- `funm_nd_∞` is typically much slower.
- `funm_nd` is typically not much slower than `funm`.
- `funm_nd` is much slower than `funm` on matrices with **close/repeated** e'vals.

The Matrix Mittag–Leffler Function

- Series definition of two-parameter matrix Mittag–Leffler (ML) function:

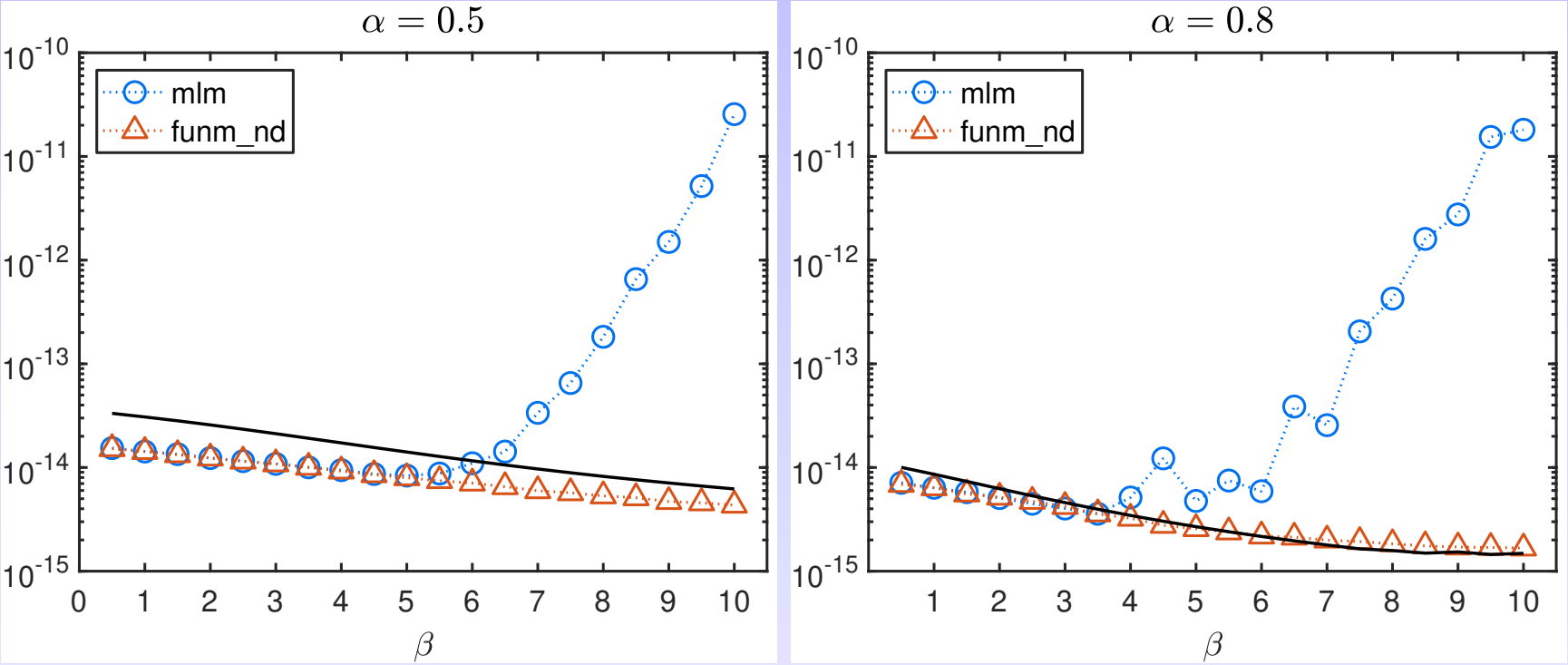
$$E_{\alpha,\beta}(A) = \sum_{k=0}^{\infty} \frac{A^k}{\Gamma(\alpha k + \beta)}, \quad \alpha, \beta \in \mathbb{C},\ \operatorname{Re}\alpha > 0,$$

where $\Gamma(\cdot)$ is the Euler gamma function.

- Applications in FDEs: $0 < \alpha < 1$ and $\beta > 0$ often.

- `mlm`, computes the derivatives and invokes the Schur–Parlett scheme (Garrappa & Popolizio, 2018).

Experimental Results on the Matrix ML Function



Relative errors in the computed $E_{\alpha,\beta}(-R)$ for the Redheffer matrix of size 20×20 and different α and β.

- The Redheffer matrix R (Barrett & Jarvis, 1992):
  - is square with $r_{ij} = 1$ if i divides j or if $j = 1$ and otherwise $r_{ij} = 0$; and
  - has $n - \lfloor \log_2 n \rfloor - 1$ e'vals s equal to 1.

Summary

- The multiprecision Schur–Parlett algorithm
  - requires at most $2n^3/3$ flops in higher precision,
  - has similar accuracy to the Schur–Parlett, and
  - needs no derivatives, so greatly expands the class of readily computable functions.
  - implicitly computes the required derivatives by finite difference using higher precision?

N. J. Higham and X. Liu.
MIMS EPrint 2020.19, September 2020. Revised March 2021.

- Codes available at https://github.com/Xiaobo-Liu/mp-spalg

Thank you the SIAM Student Travel Award for supporting the presentation!

References I

J. Banks, and A. Kulkarni, and S. Mukherjee, and N. Srivastava.
Gaussian regularization of the pseudospectrum and Davies' conjecture.
ArXiv:1906.11819, Revised April 2020.

W. W. Barrett, and T. J. Jarvis.
Spectral properties of a matrix of Redheffer.
Linear Algebra Appl., 162-164(1992), pp. 673–683.

E. B. Davies.
Approximate diagonalization.
SIAM J. Matrix Anal. Appl., 29(4):1051–1064, 2007.

P. I. Davies, and N. J. Higham.
A Schur–Parlett algorithm for computing matrix functions.
SIAM J. Matrix Anal. Appl., 25(2):464–485, 2003.

References II

J. W. Demmel.
The condition number of equivalence transformations that block diagonalize matrix pencils.
SIAM J. Numer. Anal., 20(3):599–610, 1983.

R. Garrappa, and M. Popolizio.
Computing the matrix Mittag–Leffler function with applications to fractional calculus.
J. Sci. Comput., 77(2018), pp. 129–153, 2018.

N. J. Higham.
Accuracy and Stability of Numerical Algorithms.
Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second ed., 2002.

N. J. Higham.
Functions of Matrices: Theory and Computation.
Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.

References III

N. J. Higham.
The Matrix Computation Toolbox.

N. J. Higham, and X. Liu.
A multiprecision derivative-free Schur–Parlett algorithm for computing matrix functions.
MIMS EPrint 2020.19, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, September 2020. Revised March 2021. 22 pp.

Multiprecision Computing Toolbox for MATLAB.
Advanpix, Tokyo.