



MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG



COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

A scaling and recovering algorithm for the matrix φ -functions

Xiaobo Liu

MPI for Dynamics of Complex Technical Systems, Germany

**Matrix Theory Seminar,
University of Nevada, Reno (Virtual)
October 17, 2025**

Joint work with **Awad H. Al-Mohy**
King Khalid University, KSA



1. **Background on matrix functions $f(A)$**
 - Some brief history
 - Definitions of $f(A)$
2. **The matrix exponential and φ -functions**
 - Applications in ODEs and exponential integrators
3. **Computing the matrix φ -functions**
 - From a single exponentiation
 - Scaling and scaling (recovering) method
 - Truncation error analysis
 - Cost-optimized evaluation scheme
4. **Numerical experiments**
5. **Conclusions**



1. Background on matrix functions $f(A)$

- Some brief history
- Definitions of $f(A)$

2. The matrix exponential and φ -functions

- Applications in ODEs and exponential integrators

3. Computing the matrix φ -functions

- From a single exponentiation
- Scaling and scaling (recovering) method
- Truncation error analysis
- Cost-optimized evaluation scheme

4. Numerical experiments

5. Conclusions



- Matrix functions $f: \mathbb{C}^{n \times n} \mapsto \mathbb{C}^{n \times n}$ (not elementwise) first considered by **Arthur Cayley** (1821–1895) in

A memoir on the theory of matrices (1858)

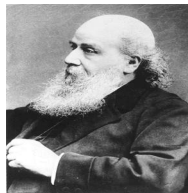
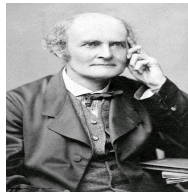
when **matrix algebra** was developed: matrix as an subject on which algebra can be performed.

- James J. Sylvester** (1814–1897) gave first definition of $f(A)$ for general f in

On the equation to the secular inequalities in the planetary theory (1883).

- Nicholas J. Higham** (1961–2024) wrote the first research monograph *fully* on matrix functions:

Functions of Matrices: Theory and Computation (2008).





[$f(A)$ via substitution] Simple substitutions, e.g.,

$$f(x) = \frac{1+x}{1-x}$$
$$\Rightarrow f(A) = (I - A)^{-1}(I + A) = (I + A)(I - A)^{-1},$$

and

$$\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots, \quad |x| < 1$$
$$\Rightarrow \log(I + A) = A - \frac{A^2}{2} + \frac{A^3}{3} - \frac{A^4}{4} + \cdots, \quad \rho(A) < 1.$$

work for f being **polynomial**, **rational function**, or **convergent power series**, but may not give a *general* definition for arbitrary A .



$$A = ZJZ^{-1} = Z \operatorname{diag}(J_1, \dots, J_p)Z^{-1}, \quad J_k = \begin{bmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{bmatrix}_{m_k \times m_k}$$

$$\Rightarrow [f(A) \text{ via JCF}] \quad f(A) := Z f(J) Z^{-1} = Z \operatorname{diag}(f(J_1), \dots, f(J_p)) Z^{-1},$$

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

- Differentiability requirement depends on the size of J_k (Higham, '08).



$$A = ZJZ^{-1} = Z \operatorname{diag}(J_1, \dots, J_p)Z^{-1}, \quad J_k = \begin{bmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{bmatrix}_{m_k \times m_k}$$

$$\Rightarrow [f(A) \text{ via JCF}] \quad f(A) := Zf(J)Z^{-1} = Z \operatorname{diag}(f(J_1), \dots, f(J_p))Z^{-1},$$

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

- Differentiability requirement depends on the size of J_k (**Higham, '08**).



[Hermite interpolation poly.] Distinct e'vals $\lambda_1, \dots, \lambda_s$, n_i max size of Jordan blocks for λ_i . Then $f(A) = p(A)$, where p is *unique* Hermite interpolating poly. with $\deg(p) < \sum_{i=1}^s n_i$ satisfying **(Higham, '08)**

$$p^j(\lambda_i) = f^{(j)}(\lambda_i), \quad j = 0:n_i - 1, \quad i = 1:s.$$

- Interpolation at the e'vals and the deriv's, corresp. to the Jordan blocks
- $\Rightarrow f(A)$ is a polynomial in A , with coeff. depending on A

[Cauchy integral formula] For $A \in \mathbb{C}^{n \times n}$,

$$f(A) = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1} dz,$$

where f is analytic on and inside a closed contour Γ that encloses $\Lambda(A)$.

▷ Equivalence of definitions, modulo analyticity assumption for Cauchy integral formula.



[Hermite interpolation poly.] Distinct e'vals $\lambda_1, \dots, \lambda_s$, n_i max size of Jordan blocks for λ_i . Then $f(A) = p(A)$, where p is *unique* Hermite interpolating poly. with $\deg(p) < \sum_{i=1}^s n_i$ satisfying (Higham, '08)

$$p^j(\lambda_i) = f^{(j)}(\lambda_i), \quad j = 0:n_i - 1, \quad i = 1:s.$$

- Interpolation at the e'vals and the deriv's, corresp. to the Jordan blocks
- $\Rightarrow f(A)$ is a polynomial in A , with coeff. depending on A

[Cauchy integral formula] For $A \in \mathbb{C}^{n \times n}$,

$$f(A) = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1} dz,$$

where f is analytic on and inside a closed contour Γ that encloses $\Lambda(A)$.

▷ Equivalence of definitions, modulo analyticity assumption for Cauchy integral formula.



[Jordan Canonical Form]: Computationally interesting for normal A (so diagonalizable).

[Hermite interpolation poly.]: Basic properties, e.g., $f(A)A = Af(A)$ and $f(A^T) = f(A)^T$.

[Cauchy integral formula]: Various uses, including quadrature methods, e.g., computing $f(A)b$ without forming $f(A)$ explicitly:

$$f(A)b = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1}b \, dz.$$

▷ Best methods often specific to particular f and A .



1. Background on matrix functions $f(A)$
 - Some brief history
 - Definitions of $f(A)$
2. **The matrix exponential and φ -functions**
 - Applications in ODEs and exponential integrators
3. Computing the matrix φ -functions
 - From a single exponentiation
 - Scaling and scaling (recovering) method
 - Truncation error analysis
 - Cost-optimized evaluation scheme
4. Numerical experiments
5. Conclusions



The most studied matrix function by far is the matrix exponential

$$e^A = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots = \sum_{k=0}^{\infty} \frac{A^k}{k!};$$

see (Moler & Van Loan; '78, '03)

Nineteen dubious ways to compute the exponential of a matrix (1978); —, twenty-five years later (2003).

Closely related are the matrix φ -functions

$$\varphi_0(A) = e^A, \quad \varphi_j(A) = \sum_{k=0}^{\infty} \frac{A^k}{(k+j)!}.$$

▷ φ_j satisfy the recurrence relation

$$\varphi_j(A) = A\varphi_{j+1}(A) + \frac{1}{j!}I.$$



For $A \in \mathbb{C}^{n \times n}$, $y \in \mathbb{C}^n$, and $y(0) = y_0$ (Minchev & Wright, '05),

$$\frac{dy}{dt} = Ay \quad \Rightarrow \quad y(t) = e^{tA} y_0,$$

$$\frac{dy}{dt} = Ay + b + ct \quad \Rightarrow \quad y(t) = e^{tA} y_0 + t\varphi_1(tA)b + t^2\varphi_2(tA)c,$$

$$\vdots$$

$$\frac{dy}{dt} = Ay + g(t, y) \quad \Rightarrow \quad y(t) = e^{tA} y_0 + \sum_{k=1}^{\infty} \varphi_k(tA) t^k g^{(k-1)}(0, y_0).$$

- A suitable truncation forms the basis of many exponential integrators.

The (nonautonomous) exponential Rosenbrock–Euler method (Pope, '63):

$$y_{n+1} = y_n + h\varphi_1(hJ_n)F(t_n, y_n) + h^2\varphi_2(hJ_n)\frac{\partial F}{\partial t}(t_n, y_n), \quad J_n = \frac{\partial F}{\partial y}(t_n, y_n),$$

where $F(t, y) \equiv Ay + g(t, y)$, $y_n \approx y(t_n)$, $t_n = nh$, and $h > 0$ stepsize.



For $A \in \mathbb{C}^{n \times n}$, $y \in \mathbb{C}^n$, and $y(0) = y_0$ (Minchev & Wright, '05),

$$\frac{dy}{dt} = Ay \quad \Rightarrow \quad y(t) = e^{tA} y_0,$$

$$\frac{dy}{dt} = Ay + b + ct \quad \Rightarrow \quad y(t) = e^{tA} y_0 + t\varphi_1(tA)b + t^2\varphi_2(tA)c,$$

$$\vdots$$

$$\frac{dy}{dt} = Ay + g(t, y) \quad \Rightarrow \quad y(t) = e^{tA} y_0 + \sum_{k=1}^{\infty} \varphi_k(tA) t^k g^{(k-1)}(0, y_0).$$

- A suitable truncation forms the basis of many exponential integrators.

The (nonautonomous) exponential Rosenbrock–Euler method (Pope, '63):

$$y_{n+1} = y_n + h\varphi_1(hJ_n)F(t_n, y_n) + h^2\varphi_2(hJ_n)\frac{\partial F}{\partial t}(t_n, y_n), \quad J_n = \frac{\partial F}{\partial y}(t_n, y_n),$$

where $F(t, y) \equiv Ay + g(t, y)$, $y_n \approx y(t_n)$, $t_n = nh$, and $h > 0$ stepsize.



Different types of exponential integrator schemes can be expressed as

$$\varphi_0(A)w_0 + \varphi_1(A)w_1 + \cdots + \varphi_p(A)w_p,$$

where w_j some vector, p relates to the exponential integrator order.

[Compute $\varphi_j(A)w_j$ by Krylov method]:

1. Initialize $v_1 = w_j / \|w_j\|_2$.
2. Build orthonormal basis $V_m = [v_1, v_2, \dots, v_m]$ of $\mathcal{K}_m(A, w_j) = \text{span}\{w_j, Aw_j, \dots, A^{m-1}w_j\}$ via the Arnoldi process:

$$AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T.$$

3. Project $\varphi_j(A)w_j$ onto a lower-dim. subspace (Hochbruck & Lubich, '97):

$$\varphi_j(A)w_j \approx \frac{\|w_j\|_2}{2\pi i} \int_{\Gamma} \varphi_j(\sigma) V_m (\sigma I - H_m)^{-1} e_1 d\sigma = \|w_j\|_2 V_m \varphi_j(H_m) e_1.$$

▷ Accurately and stably computing $\varphi_j(H_m)$ a key and challenging step.



1. Background on matrix functions $f(A)$
 - Some brief history
 - Definitions of $f(A)$
2. The matrix exponential and φ -functions
 - Applications in ODEs and exponential integrators
3. **Computing the matrix φ -functions**
 - From a single exponentiation
 - Scaling and scaling (recovering) method
 - Truncation error analysis
 - Cost-optimized evaluation scheme
4. Numerical experiments
5. Conclusions

**Theorem (Higham, '08)**

$$B = \begin{bmatrix} A & I \\ 0 & 0 \end{bmatrix} \in \mathbb{C}^{2n \times 2n} \Rightarrow e^B = \begin{bmatrix} e^A & \varphi_1(A) \\ 0 & I \end{bmatrix}.$$

Theorem (Al-Mohy & L., '25)

Let $E = \begin{bmatrix} I & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n \times np}$ and $J = J_p(0) \otimes I$, where $J_p(0)$ is the $p \times p$ Jordan block associated with zero. Then

$$W = \begin{bmatrix} A & E \\ 0 & J \end{bmatrix} \in \mathbb{C}^{(p+1)n \times (p+1)n} \Rightarrow e^W = \begin{bmatrix} e^A & \varphi_1(A) & \cdots & \varphi_p(A) \\ 0 & e^{J_p(0)} \otimes I & & \end{bmatrix}.$$

- Theorem proved for arbitrary analytic f : the first block row of $f(W)$ is

$$[f(W)]_{1:n, 1:(p+1)n} = [g_0(A) \quad g_1(A) \quad g_2(A) \quad \cdots \quad g_p(A)],$$

where $g_i(A) = Ag_{i+1}(A) + f^{(i)}(0)I/i!$, $g_0 = f$, $i = 0:p-1$.

**Theorem (Higham, '08)**

$$B = \begin{bmatrix} A & I \\ 0 & 0 \end{bmatrix} \in \mathbb{C}^{2n \times 2n} \quad \Rightarrow \quad e^B = \begin{bmatrix} e^A & \varphi_1(A) \\ 0 & I \end{bmatrix}.$$

Theorem (Al-Mohy & L., '25)

Let $E = \begin{bmatrix} I & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n \times np}$ and $J = J_p(0) \otimes I$, where $J_p(0)$ is the $p \times p$ Jordan block associated with zero. Then

$$W = \begin{bmatrix} A & E \\ 0 & J \end{bmatrix} \in \mathbb{C}^{(p+1)n \times (p+1)n} \quad \Rightarrow \quad e^W = \begin{bmatrix} e^A & \varphi_1(A) & \cdots & \varphi_p(A) \\ 0 & e^{J_p(0)} \otimes I & & \end{bmatrix}.$$

- Theorem proved for arbitrary analytic f : the first block row of $f(W)$ is

$$[f(W)]_{1:n, 1:(p+1)n} = \begin{bmatrix} g_0(A) & g_1(A) & g_2(A) & \cdots & g_p(A) \end{bmatrix},$$

where $g_i(A) = Ag_{i+1}(A) + f^{(i)}(0)I/i!$, $g_0 = f$, $i = 0:p-1$.

**Theorem (Higham, '08)**

$$B = \begin{bmatrix} A & I \\ 0 & 0 \end{bmatrix} \in \mathbb{C}^{2n \times 2n} \quad \Rightarrow \quad e^B = \begin{bmatrix} e^A & \varphi_1(A) \\ 0 & I \end{bmatrix}.$$

Theorem (Al-Mohy & L., '25)

Let $E = \begin{bmatrix} I & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n \times np}$ and $J = J_p(0) \otimes I$, where $J_p(0)$ is the $p \times p$ Jordan block associated with zero. Then

$$W = \begin{bmatrix} A & E \\ 0 & J \end{bmatrix} \in \mathbb{C}^{(p+1)n \times (p+1)n} \quad \Rightarrow \quad e^W = \begin{bmatrix} e^A & \varphi_1(A) & \cdots & \varphi_p(A) \\ 0 & e^{J_p(0)} \otimes I & & \end{bmatrix}.$$

- Theorem proved for **arbitrary analytic** f : the first block row of $f(W)$ is

$$[f(W)]_{1:n, 1:(p+1)n} = \begin{bmatrix} g_0(A) & g_1(A) & g_2(A) & \cdots & g_p(A) \end{bmatrix},$$

where $g_i(A) = Ag_{i+1}(A) + f^{(i)}(0)I/i!$, $g_0 = f$, $i = 0:p-1$.



$\mathcal{R}_m^{(p)}(z) := N_m(z)/D_m(z)$ known explicitly (**Berland, Skaflestad, Wright; '07**), (**Higham, '08**):

$$N_m(z) = \frac{m!}{(2m+p)!} \sum_{i=0}^m \left[\sum_{j=0}^i \frac{(2m+p-j)!(-1)^j}{j!(m-j)!(p+i-j)!} \right] z^i,$$
$$D_m(z) = \frac{m!}{(2m+p)!} \sum_{i=0}^m \frac{(2m+p-i)!}{i!(m-i)!} (-z)^i.$$

Padé truncation (approximation) error satisfies

$$\varphi_p(z) - \mathcal{R}_m^{(p)}(z) = \frac{(-1)^m(m+p)!m!}{(2m+p)!(2m+p+1)!} z^{2m+1} + O(z^{2m+2}).$$



Theorem (Al-Mohy & L., '25)

The $[m + p - j / m]$ Padé approximants to $\varphi_j(z)$, $\mathcal{R}_{m+p-j,m}(z) =: \mathcal{R}_m^{(j)}(z)$, $j = 1 : p$, satisfy the recurrence relation

$$\mathcal{R}_m^{(j)}(z) = z\mathcal{R}_m^{(j+1)}(z) + \frac{1}{j!}, \quad j = p-1 : -1 : 0.$$

Idea of proof: set f as the $[m + p / m]$ Padé approximant, $\mathcal{R}_m^{(0)}(z)$, to e^z .

- Adapt the Padé approximant degree per φ_j and **fix the denominator** in the construction.

$\rightsquigarrow$ Comput'nal advantage over evaluating fixed-deg. Padé approximants to φ -functions *independently*, avoiding **repeated** rational approximations.



For the matrix exponential: $e^A = (e^{2^{-s}A})^{2^s} \approx r_m(2^{-s}A)^{2^s}$

-
- | | |
|---|------------|
| 1 $B \leftarrow A/2^s$ so $\ B\ _1$ is close to the origin. | ▷ scaling |
| 2 Evaluate the $[m/m]$ Padé approximant $r_m(B)$ to e^B . | ▷ evaluate |
| 3 $e^A \approx r_m(B)^{2^s}$ | ▷ squaring |
-

- Proposed by Lawson (**Lawson, '67**).
- Algorithm, with rounding error analysis and a posteriori error bound (**Ward, '77**).
- Backward truncation error analysis allowing choice of s and m (**Moler & Van Loan, '78**).
- Sharper b'ward trunc. analysis giving optimal s and m minimizing computational cost in double precision (**Higham, '05**).
- B'ward trunc. bound based on $\|A^k\|^{1/k}$ rather than $\|A\|$, alleviating overscaling issues (**Al-Mohy & Higham, '09**), current MATLAB expm.



Simultaneously for the φ -functions: $\varphi_0(A) = e^A$, $\varphi_1(A)$, $\dots$, $\varphi_p(A)$

- 1 Select a scaling parameter s and a degree m of Padé approximant.
 - 2 $A \leftarrow A/2^s$
 - 3 Evaluate the $[m/m]$ Padé approximant, $\mathcal{R}_m^{(p)}(A)$, to $\varphi_p(A)$.
 - 4 Invoke the recurrence $\mathcal{R}_m^{(j)}(A) = A\mathcal{R}_m^{(j+1)}(A) + \frac{1}{j!}I$, $j = p-1:-1:0$.
 - 5 **for** $i = 1:s$ **do**
 - 6 **for** $j = p:-1:0$ **do**
 - 7 $\mathcal{R}_m^{(j)}(A) \leftarrow 2^{-j} \left(\mathcal{R}_m^{(0)}(A)\mathcal{R}_m^{(j)}(A) + \sum_{k=1}^j \mathcal{R}_m^{(k)}(A)/(j-k)! \right)$
 - 8 **end**
 - 9 **end**
-

- The **double-argument formula** $\varphi_j(2A) = \frac{1}{2^j} \left(\varphi_0(A)\varphi_j(A) + \sum_{k=1}^j \frac{\varphi_k(A)}{(j-k)!} \right)$ (Berland, Skaflestad, Wright; '07) for recov. $\varphi_j(A)$ from $\varphi_j(2^{-s}A)$, $j = 0:p$.
 $\rightsquigarrow$ Determine s and m to *guarantee accuracy* and *minimize comput'nal cost*.



Key: Set f as the $[m + p/m]$ Padé approximant, $\mathcal{R}_m^{(0)}(z)$, to e^z :

$$W = \begin{bmatrix} A & E \\ 0 & J \end{bmatrix} \Rightarrow \mathcal{R}_m^{(0)}(W) = \begin{bmatrix} \mathcal{R}_m^{(0)}(A) & \mathcal{R}_m^{(1)}(A) \dots \mathcal{R}_m^{(p)}(A) \\ 0 & \mathcal{R}_m^{(0)}(J) \end{bmatrix}.$$

- **Evaluation** starts from the $[m/m]$ Padé approximant, $\mathcal{R}_m^{(p)}(A)$, to $\varphi_p(A)$, **truncation error analysis** based on $\mathcal{R}_m^{(0)}(A)$ to e^A . Let (**Higham, '08**)

$$h_{m,p}(A) = \log(e^{-A} \mathcal{R}_m^{(0)}(A)) = \sum_{k=2m+p+1}^{\infty} c_{m,p,k} A^k,$$

then $\mathcal{R}_m^{(0)}(A) = e^{A+h_{m,p}(A)}$. Hence

$$\left[\mathcal{R}_m^{(0)}(2^{-s} A) \right]^{2^s} = e^{A+2^s h_{m,p}(2^{-s} A)} =: e^{A+\Delta A},$$

and $\Delta A = 2^s h_{m,p}(2^{-s} A)$ represents the **backward error** resulting from approximating e^A via scaling and squaring.



The selected s need satisfy

$$\frac{\|\Delta A\|_1}{\|A\|_1} = \frac{2^s \|h_{m,p}(2^{-s}A)\|_1}{\|A\|_1} = \frac{\|h_{m,p}(2^{-s}A)\|_1}{\|2^{-s}A\|_1} \leq u \approx 1.1 \times 10^{-16}.$$

Computing *implicitly* $\exp \left(\begin{bmatrix} A + \Delta A & E + \Delta E \\ 0 & J + \Delta J \end{bmatrix} \right)$, what about ΔE ?

Use the linear operator $\mathcal{D}_f(A, B, E)$ in (Al-Mohy, '25):

$$f \left(\begin{bmatrix} A & E \\ 0 & B \end{bmatrix} \right) = \begin{bmatrix} f(A) & \mathcal{D}_f(A, B, E) \\ 0 & f(B) \end{bmatrix}.$$

$$\Delta E = \mathcal{D}_{h_{m,p}}(2^{-s}A, 2^{-s}J, E) = (1, 2) \text{ block of } 2^s h_{m,p} \left(\begin{bmatrix} 2^{-s}A & 2^{-s}E \\ 0 & 2^{-s}J \end{bmatrix} \right).$$

$\rightsquigarrow$ Invoke the recurrence with $f = 2^s h_{m,p}$.



The j th block of ΔE is

$$(\Delta E)_j = 2^s A^{-j} h_{m,p}(2^{-s} A), \quad 1 \leq j \leq p.$$

Therefore, select s such that

$$\frac{\|\Delta E\|_1}{\|E\|_1} = \|\Delta E\|_1 \leq \max_{1 \leq j \leq p} \|(2^{-s} A)^{-j} h_{m,p}(2^{-s} A)\|_1 \leq \max_{1 \leq j \leq p} \frac{\tilde{h}_{m,p}(\theta)}{\theta^j} \leq u.$$

$$\tilde{h}_{m,p}(\theta) = \sum_{k=2m+p+1}^{\infty} |c_{m,p,k}| \theta^k, \quad \theta \geq 0 \text{ real-valued function of } 2^{-s} A.$$

- A simple choice for θ is $\|2^{-s} A\|_1$ (**Higham, '08**)

$$\Rightarrow \left\| \sum_{k=2m+p+1}^{\infty} |c_{m,p,k}| (2^{-s} A)^{k-j} \right\| \leq \sum_{k=2m+p+1}^{\infty} |c_{m,p,k}| \|2^{-s} A\|_1^{k-j}.$$

- **Idea:** symbolically compute $c_{m,p,k}$ and find the **scaling threshold**

$$\theta_{m,p} = \max \left\{ \theta : \max_{1 \leq j \leq p} \tilde{h}_{m,p}(\theta) / \theta^j \leq u \approx 1.1 \times 10^{-16} \right\}.$$



Using $\theta = \alpha_r(2^{-s}A)$ (not $\|2^{-s}A\|_1$) leads to sharper bound (Al-Mohy & Higham, '09):

$$\alpha_r(A) = \min_{2 \leq r \leq r_{\max}} \max \left(\|A^r\|_1^{1/r}, \|A^{r+1}\|_1^{1/(r+1)} \right).$$

• Motivation:

$$\rho(A) \leq \|A^r\|_1^{1/r} \leq \|A\|, \quad r = 1 : \infty,$$

$\rho(A) \ll \|A\|$ for highly non-normal A , and $\lim_{r \rightarrow \infty} \|A^r\|_1^{1/r} = \rho(A)$.

$\rightsquigarrow \|A^r\|_1^{1/r}$ instead of $\|A\|$ in bounding $\left\| \sum_{k=2m+p+1}^{\infty} |c_{m,p,k}| (2^{-s}A)^{k-j} \right\|$.

The overall relative backward errors are bounded by

$$\max \left(\frac{\|\Delta A\|_1}{\|A\|_1}, \frac{\|\Delta E\|_1}{\|E\|_1} \right) \leq \frac{\tilde{h}_{m,p}(\alpha_r(2^{-s}A))}{\alpha_r(2^{-s}A)^\delta} \leq u, \quad \delta = 1 \text{ or } p.$$

• Need choose optimal s and m s.t. $2^{-s}\alpha_r(A) \leq \theta_{m,p}$ for $2 \leq r \leq r_{\max}$.

**Table:** Selected values of $\theta_{m_i,p}$, at optimal Paterson–Stockmeyer degrees m_i .

p	$m_0 = 1$	$m_1 = 2$	$m_2 = 3$	$m_3 = 4$	$m_4 = 6$	$m_5 = 8$	$m_6 = 10$	$m_7 = 12$
1	2.00e-5	3.81e-3	3.97e-2	1.54e-1	7.26e-1	1.76	3.17	4.87
2	3.76e-5	6.09e-3	5.81e-2	2.13e-1	9.28e-1	2.06	3.54	5.28
3	7.37e-5	9.87e-3	8.53e-2	2.94e-1	1.16	2.37	3.91	5.69
4	1.50e-4	1.62e-2	1.26e-1	4.06e-1	1.40	2.69	4.28	6.09
5	3.15e-4	2.70e-2	1.87e-1	5.62e-1	1.66	3.01	4.65	6.50
6	6.86e-4	4.55e-2	2.80e-1	7.79e-1	1.92	3.34	5.02	6.90
7	1.54e-3	7.75e-2	4.18e-1	1.05	2.20	3.68	5.40	7.30
8	3.54e-3	1.33e-1	6.26e-1	1.26	2.48	4.01	5.77	7.69
9	8.35e-3	2.30e-1	9.34e-1	1.48	2.77	4.35	6.14	8.08
10	2.01e-2	3.99e-1	1.16	1.71	3.07	4.69	6.51	8.47

- Scale A down by 2^s such that $2^{-s}\alpha_r(A) \leq \theta_{m,p}$. Larger $m \rightsquigarrow$ higher threshold $\theta_{m,p}$ and less strong scaling needed.

Next: choose *feasible* s and m to minimize the computational cost.



Evaluate the $[m/m]$ Padé approximant, $\mathcal{R}_m^{(p)}(A)$, to $\varphi_p(A)$ by the PS scheme:

$$q(A) = \sum_{k=0}^{\nu} B_k(A) \cdot (A^\tau)^k, \quad \nu = \left\lfloor \frac{m}{\tau} \right\rfloor,$$

where the polynomial q denotes either N_m or D_m ,

$$B_k(A) = \begin{cases} \sum_{j=0}^{\tau-1} c_{\tau k+j} A^j, & k = 0, 1, \dots, \nu - 1, \\ \sum_{j=0}^{m-\tau\nu} c_{\tau\nu+j} A^j, & k = \nu. \end{cases}$$

- Idea: For parameter $\tau \in \mathbb{N}^+$, rewrite $q(A)$ as a polynomial in A^τ with **matrix coefficients** $B_k(A)$. For example, with $m = 6$ and $\tau = 3$,

$$q(A) = \underbrace{c_6 I}_{B_2(A)} (A^3)^2 + \underbrace{(c_5 A^2 + c_4 A + c_3 I)}_{B_1(A)} A^3 + \underbrace{(c_2 A^2 + c_1 A + c_0 I)}_{B_0(A)}.$$



Comput'nal cost characterized by the **equivalent number of matrix multiplications**.

The number of mat multip'ns required to simultaneously evaluate N_m and D_m is minimized at $\tau_* = \lfloor \sqrt{2m} \rfloor$ or $\tau_* = \lceil \sqrt{2m} \rceil$. The optimal degrees given by **(Fasi, '19)**

$$m_i := \left\lfloor \frac{(i+3)^2}{8} \right\rfloor, \quad i = 0, 1, 2, \dots$$

- With same number of mat multip'ns i , the highest degree m_i is optimal.



- 1 Select a scaling parameter s and a degree m of Padé approximant.
- 2 $A \leftarrow A/2^s$
- 3 Evaluate the $[m/m]$ Padé approximant, $\mathcal{R}_m^{(p)}(A)$, to $\varphi_p(A)$.
- 4 Invoke the recurrence $\mathcal{R}_m^{(j)}(A) = A\mathcal{R}_m^{(j+1)}(A) + \frac{1}{j!}I$, $j = p-1:-1:0$.
- 5 **for** $i = 1:s$ **do**
- 6 **for** $j = p:-1:0$ **do**
- 7 $\mathcal{R}_m^{(j)}(A) \leftarrow 2^{-j} \left(\mathcal{R}_m^{(0)}(A)\mathcal{R}_m^{(j)}(A) + \sum_{k=1}^j \mathcal{R}_m^{(k)}(A)/(j-k)! \right)$
- 8 **end**
- 9 **end**

Total cost is

$$C_{m_i,s} = i + p + \frac{4}{3} + s(p+1),$$

where $s = \max(\lceil \log_2(\alpha_r(A)/\theta_{m,p}) \rceil, 0)$ s.t. $2^{-s}\alpha_r(A) \leq \theta_{m,p}$.

$\rightsquigarrow$ Cost minimized over the index pair (m_i, r) , $m_i \leq m_{\max} = 12$ s.t.
 $\kappa(D_{m_i}(A))$ modest, $2 \leq r \leq r_{\max}$, $r_{\max}$ determined when $m_{\max}$ fixed.



1. Background on matrix functions $f(A)$
 - Some brief history
 - Definitions of $f(A)$
2. The matrix exponential and φ -functions
 - Applications in ODEs and exponential integrators
3. Computing the matrix φ -functions
 - From a single exponentiation
 - Scaling and scaling (recovering) method
 - Truncation error analysis
 - Cost-optimized evaluation scheme
4. **Numerical experiments**
5. Conclusions



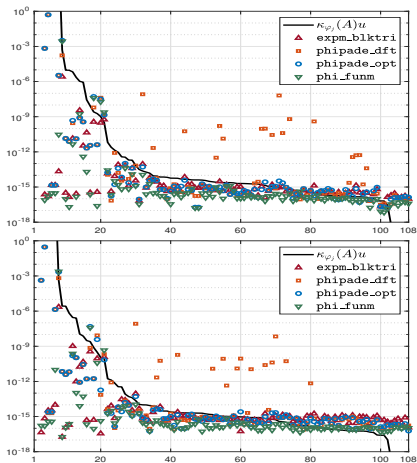
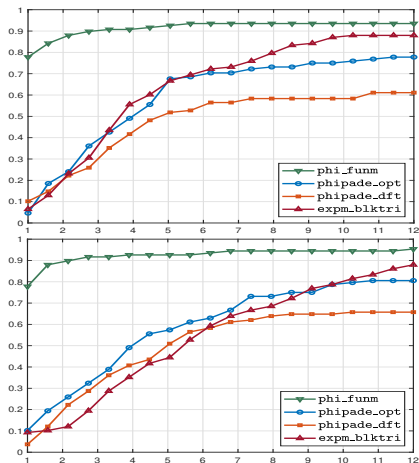
Algorithms:

- **phi_funm**: the proposed algorithm;
- **phipade**: from EXPINT (Berland, Skaflestad, Wright; '07), realizing the algorithm (Skaflestad & Wright, '09) executed in two configurations:
 - **phipade_dft**: the default setting: uses the [7/7] Padé approximant;
 - **phipade_opt**: the adaptive setting proposed in (Skaflestad & Wright, '09), seeking the optimal Padé degree $3 \leq m \leq 13$ to minimize the cost;
- **expm_blktri**: the algorithm of (Al-Mohy, '25), for computing the exp of block triangular matrices. (Comput. e^W and extract (1,2) block.)

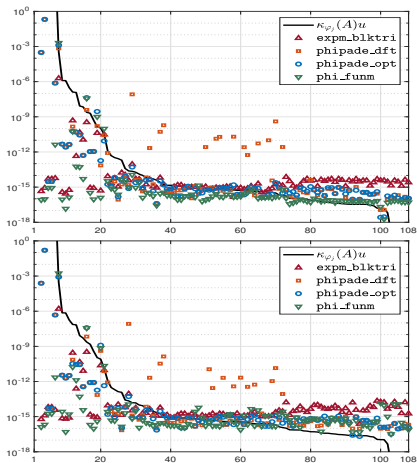
Test set:

- 108 *nonnormal* matrices from matrix function literature, $2 \leq n \leq 41$.
- 5 Hessenberg matrices constructed from Arnoldi within Krylov methods for computing $\varphi_j(A)e$, $e = \text{ones}(n,1)$, $900 \leq n \leq 392,257$.

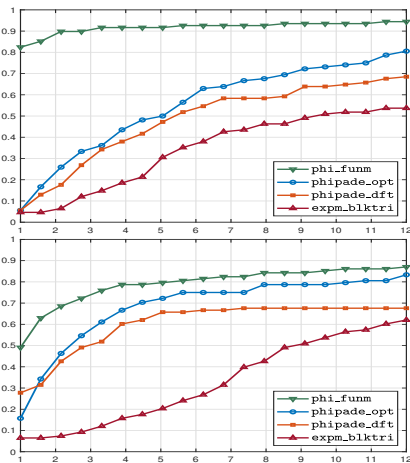
Working precision: binary64 (IEEE double) in MATLAB

Top: $j = 1$; bottom: $j = 4$.

- $p = 10$. Left: 1-norm forward error. Right: performance profiles: fraction of tests whose relative error is within factor (x -axis) of the best.

Top: $j = 7$; bottom: $j = 10$.

- `phipade_dft` unstable (overscaling), `expm_blktri` worsen as j grows.



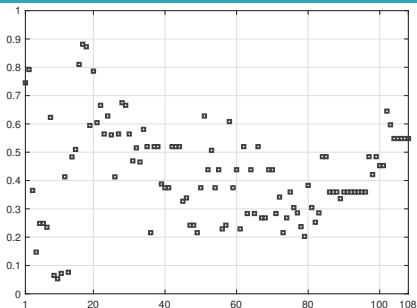
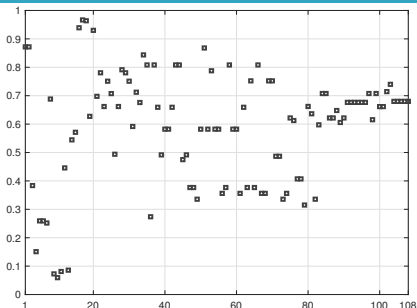
(a) `phi_funm` over `phipade_dft`.(b) `phi_funm` over `phipade_opt`.

Figure: Ratios of computational cost: `phi_funm` to `phipade`.

- `phi_funm` *always* more efficient, can be $10\times$ to $20\times$ cheaper:
 - based on $\alpha_r(A)$ rather than $\|A\|_1 \rightsquigarrow$ less prone to overscaling.
 - adapts the Padé degree per $\varphi_j \rightsquigarrow$ saves at least p multiple linear-system solvers.

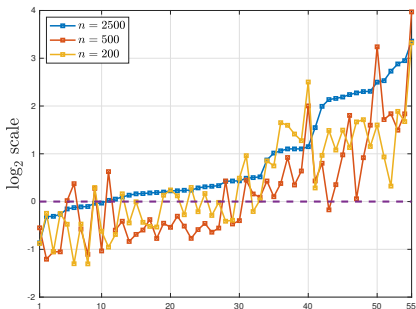
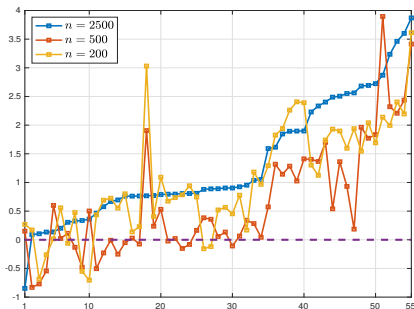
(a) `phi_funm` vs `phipade_dft`.(b) `phi_funm` vs `phipade_opt`.

Figure: $\log_2$ -speedup of `phi_funm` to `phipade`. $y \geq 0$ indicate `phi_funm` is faster.

- For $n = O(100)$, `phi_funm` and `phipade` comparable in speed.
- Lower asymptotic cost of `phi_funm` reflected in runtime as n grows.
- `phi_funm` more *reliable*: runtimes $\lesssim 2\times$ the fastest, $16\times$ for `phipade`.



	n	M_{eval}	M_{recv}	P_{par}	P_{eval}	P_{recv}	T_{tot}
A = <code>circul</code>	20	17	55	52.0%	38.1%	9.9%	0.0
	200	16	132	3.1%	9.5%	87.4%	0.1
	500	16	165	2.5%	9.8%	87.7%	0.6
	2500	17	209	0.8%	6.4%	92.8%	44.2
B = <code>triw</code>	20	16	11	76.1%	13.9%	10.0%	0.0
	200	16	55	8.7%	16.0%	75.2%	0.0
	500	16	66	1.9%	18.1%	80.0%	0.3
	2500	17	88	1.9%	13.3%	84.8%	20.6
C = <code>smoke</code>	20	14	0	70.8%	29.2%	0.0%	0.0
	200	14	0	22.8%	77.2%	0.0%	0.0
	500	14	0	7.4%	92.6%	0.0%	0.2
	2500	14	0	7.2%	92.8%	0.0%	8.3

- $\|A\|_1 = n(n+1)/2$, $\|B\|_1 = 2n-1$ (increasing norms): number of mat multip'ns in **recovery phase** increases with n . $\|C\|_1 = 2$: no scaling.
- $O(n^2)$ **parameter selection**: costly for small n , negligible as n increases.
- P_{eval} , P_{recv} scale more consist. with M_{eval} and M_{recv} , resp., as n grows.



		bcspwr10		gr_30_30		helm2d03		orani678		poisson99	
$p = 1$		Error	Cost	Error	Cost	Error	Cost	Error	Cost	Error	Cost
$m = 30$	phi_funm	3.2e-15	11.3	1.0e-15	12.3	1.4e-15	12.3	3.7e-16	8.3	7.5e-14	34.3
	phipade_dft	5.3e-9	14.7	1.1e-9	16.7	1.9e-10	16.7	5.7e-16	14.7	8.2e-14	38.7
	phipade_opt	2.0e-15	13.7	3.3e-15	15.7	2.1e-15	15.7	7.4e-16	13.7	7.8e-14	35.7
$m = 80$	phi_funm	3.2e-15	11.3	1.0e-15	12.3	1.5e-15	12.3	4.9e-16	8.3	9.1e-14	34.3
	phipade_dft	5.3e-9	14.7	3.4e-14	18.7	1.9e-10	16.7	6.4e-16	18.7	1.1e-13	38.7
	phipade_opt	2.0e-15	13.7	1.8e-15	15.7	2.1e-15	15.7	9.8e-16	15.7	9.9e-14	35.7
$p = 4$		Error	Cost	Error	Cost	Error	Cost	Error	Cost	Error	Cost
$m = 30$	phi_funm	1.1e-15	16.3	8.2e-15	17.3	4.0e-15	17.3	6.8e-16	10.3	1.5e-14	72.3
	phipade_dft	5.0e-10	27.7	1.1e-9	32.7	1.8e-10	32.7	3.6e-16	27.7	5.4e-14	87.7
	phipade_opt	1.3e-15	23.7	3.1e-15	28.7	1.9e-15	28.7	4.5e-16	23.7	5.2e-14	78.7
$m = 80$	phi_funm	1.1e-15	16.3	8.8e-15	17.3	4.1e-15	17.3	8.6e-16	10.3	2.0e-14	72.3
	phipade_dft	5.0e-10	27.7	3.3e-14	37.7	1.8e-10	32.7	1.3e-15	37.7	6.4e-14	87.7
	phipade_opt	1.3e-15	23.7	1.6e-15	28.7	1.9e-15	28.7	1.1e-15	28.7	5.9e-14	78.7

- **phi_funm** always has lower *computational cost*. Execution times comparable between 10^{-3} and 10^{-2} seconds (not shown).
- **phi_funm** and **phipade_opt** deliver good and comparable accuracy, but **phipade_dft** again shows *instability*.



1. Background on matrix functions $f(A)$
 - Some brief history
 - Definitions of $f(A)$
2. The matrix exponential and φ -functions
 - Applications in ODEs and exponential integrators
3. Computing the matrix φ -functions
 - From a single exponentiation
 - Scaling and scaling (recovering) method
 - Truncation error analysis
 - Cost-optimized evaluation scheme
4. Numerical experiments
5. **Conclusions**



Faster, more accurate, b'ward stable algorithm for matrix φ -functions:

- New recurrence between non-diagonal Padé approximants to $\varphi_j \rightsquigarrow$ Avoid repeated rational approximations $\rightsquigarrow$ Computational savings.
- Rigorous b'ward error analysis, $\rightsquigarrow$ Sharp relative error bounds & Proposed algorithm b'ward stable in exact arithmetic.
- Algorithmic parameters selected on the fly in $O(n^2)$ to minimize the overall computational cost.
- Matrix triangularity exploitation (not shown) by recomput. diagonals of $\varphi_0(A) = e^A$ via explicit formulae $\rightsquigarrow$ Controlled error propagation.

► A. H. Al-Mohy and X. Liu. A scaling and recovering algorithm for the matrix φ -functions. SIAM J. Sci. Comput., To appear.
Currently available at <https://arxiv.org/abs/2506.01193>.

Code available at https://github.com/xiaobo-liu/phi_funm

Next?

- Excellent *numerical* f'ward stability observed. Rounding error analysis?



Faster, more accurate, b'ward stable algorithm for matrix φ -functions:

- New recurrence between non-diagonal Padé approximants to $\varphi_j \rightsquigarrow$ Avoid repeated rational approximations $\rightsquigarrow$ Computational savings.
- Rigorous b'ward error analysis, $\rightsquigarrow$ Sharp relative error bounds & Proposed algorithm b'ward stable in exact arithmetic.
- Algorithmic parameters selected on the fly in $O(n^2)$ to minimize the overall computational cost.
- Matrix triangularity exploitation (not shown) by recomput. diagonals of $\varphi_0(A) = e^A$ via explicit formulae $\rightsquigarrow$ Controlled error propagation.

► A. H. Al-Mohy and X. Liu. A scaling and recovering algorithm for the matrix φ -functions. SIAM J. Sci. Comput., To appear.
Currently available at <https://arxiv.org/abs/2506.01193>.

Code available at https://github.com/xiaobo-liu/phi_funm

Next?

- Excellent *numerical* f'ward stability observed. Rounding error analysis?



Awad H. Al-Mohy.

A new algorithm for computing the exponential of a block triangular matrix.

SIAM J. Sci. Comput., To appear.



Awad H. Al-Mohy and Nicholas J. Higham.

A new scaling and squaring algorithm for the matrix exponential.

SIAM J. Matrix Anal. Appl., 31(3):970–989, 2009.



Håvard Berland, Bård Skaflestad, and Will M. Wright.

EXPINT—A MATLAB package for exponential integrators.

ACM Trans. Math. Softw., 33(1):4–es, 2007.



Massimiliano Fasi.

Optimality of the Paterson–Stockmeyer method for evaluating matrix polynomials and rational matrix functions.

Linear Algebra Appl., 574:182–200, 2019.



Nicholas J. Higham.

The scaling and squaring method for the matrix exponential revisited.

SIAM J. Matrix Anal. Appl., 26(4):1179–1193, 2005.



Nicholas J. Higham.

Functions of Matrices: Theory and Computation.

SIAM, Philadelphia, PA, USA, 2008. xx+425 pp. ISBN 978-0-898716-46-7.



Marlis Hochbruck and Christian Lubich.

On Krylov subspace approximations to the matrix exponential operator.

SIAM J. Numer. Anal., 34(5):1911–1925, 1997.



J. Douglas Lawson.

Generalized Runge–Kutta processes for stable systems with large Lipschitz constants.

SIAM J. Numer. Anal., 4(3):372–380, 1967.



Borislav V. Minchev and Will M. Wright.

A review of exponential integrators for first order semi-linear problems.

Tech. Report 2/05, Norwegian University of Science and Technology, Trondheim, Norway, 2005.



Cleve Moler and Charles Van Loan.

Nineteen dubious ways to compute the exponential of a matrix.

SIAM Rev., 20(4):801–836, 1978.



Cleve Moler and Charles Van Loan.

Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later.
SIAM Rev., 45(1):3–49, 2003.



David A. Pope.

An exponential method of numerical integration of ordinary differential equations.
Comm. Assoc. Comput. Mach., 6(8):491–493, 1963.



Bård Skaflestad and Will M. Wright.

The scaling and modified squaring method for matrix functions related to the exponential.
Appl. Numer. Math., 59(3–4):783–799, 2009.



Robert C. Ward.

Numerical computation of the matrix exponential with accuracy estimate.
SIAM J. Numer. Anal., 14(4):600–610, 1977.