Adaptive rational Krylov methods for exponential Runge–Kutta integrators

Kai Bergermann Martin Stoll

Chemnitz University of Technology, Faculty of Mathematics,
Research Group Scientific Computing

Exploiting Algebraic and Geometric Structure in Time-Integration Methods, Pisa
Apr 04, 2024

Stiff systems of ODEs

We consider **semilinear parabolic PDEs** with the splitting

$$\frac{\partial u(\boldsymbol{x},t)}{\partial t} = F(t,u(\boldsymbol{x},t)) = -\mathcal{A}u(\boldsymbol{x},t) + g(t,u(\boldsymbol{x},t)), \quad u(\boldsymbol{x},t=0) = u_0,$$

with $\mathcal{A}$ linear differential operator (here: $\mathcal{A} = -\Delta$) between appropriate function spaces.

We are interested in *discrete* linear semi-definite differential operators, e.g.,

- finite differences: $\boldsymbol{A} = \frac{1}{h_x^2}\left(\boldsymbol{T}_{n_x} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{T}_{n_x}\right)$, $\Sigma \subseteq \frac{2}{h_x^2}[0,4]$
- discrete graph setting: $\boldsymbol{A} = \boldsymbol{L}$ graph Laplacian, $\Sigma \subseteq [0,n]$

Leads to systems of ODEs

$$\frac{\partial \boldsymbol{u}(t)}{\partial t} = -\boldsymbol{A}\boldsymbol{u}(t) + g(t,\boldsymbol{u}(t)), \quad \boldsymbol{u}(t=0) = \boldsymbol{u}_0,$$

with $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{u}\colon [0,T] \mapsto \mathbb{R}^n$.

We consider **semilinear parabolic PDEs** with the splitting

$$\frac{\partial u(\boldsymbol{x},t)}{\partial t} = F(t,u(\boldsymbol{x},t)) = -\mathcal{A}u(\boldsymbol{x},t) + g(t,u(\boldsymbol{x},t)), \quad u(\boldsymbol{x},t=0) = u_0,$$

with $\mathcal{A}$ linear differential operator (here: $\mathcal{A} = -\Delta$) between appropriate function spaces.

We are interested in *discrete* linear semi-definite differential operators, e.g.,

- finite differences: $\boldsymbol{A} = \frac{1}{h_x^2}\left(\boldsymbol{T}_{n_x} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{T}_{n_x}\right)$, $\Sigma \subseteq \frac{2}{h_x^2}[0,4]$
- discrete graph setting: $\boldsymbol{A} = \boldsymbol{L}$ graph Laplacian, $\Sigma \subseteq [0,n]$

Leads to systems of ODEs

$$\frac{\partial \boldsymbol{u}(t)}{\partial t} = -\boldsymbol{A}\boldsymbol{u}(t) + g(t,\boldsymbol{u}(t)), \quad \boldsymbol{u}(t=0) = \boldsymbol{u}_0,$$

with $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{u}\colon [0,T] \mapsto \mathbb{R}^n$.

We consider **semilinear parabolic PDEs** with the splitting

$$\frac{\partial u(\boldsymbol{x},t)}{\partial t} = F(t,u(\boldsymbol{x},t)) = -\mathcal{A}u(\boldsymbol{x},t) + g(t,u(\boldsymbol{x},t)), \quad u(\boldsymbol{x},t=0) = u_0,$$

with $\mathcal{A}$ linear differential operator (here: $\mathcal{A} = -\Delta$) between appropriate function spaces.

We are interested in *discrete* linear semi-definite differential operators, e.g.,

- finite differences: $\boldsymbol{A} = \frac{1}{h_x^2}(\boldsymbol{T}_{n_x} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{T}_{n_x})$, $\Sigma \subseteq \frac{2}{h_x^2}[0,4]$
- discrete graph setting: $\boldsymbol{A} = \boldsymbol{L}$ graph Laplacian, $\Sigma \subseteq [0,n]$

Leads to systems of ODEs

$$\frac{\partial \boldsymbol{u}(t)}{\partial t} = -\boldsymbol{A}\boldsymbol{u}(t) + g(t,\boldsymbol{u}(t)), \quad \boldsymbol{u}(t=0) = \boldsymbol{u}_0,$$

with $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{u}\colon [0,T] \mapsto \mathbb{R}^n$.

Exponential integrators

- Starting point: variation-of-constants formula [Hochbruck, Ostermann, 2010]

$$\boldsymbol{u}(t) = e^{-t\boldsymbol{A}}\boldsymbol{u}_0 + \int_0^t e^{-(t-\tau)\boldsymbol{A}} g(\tau, \boldsymbol{u}(\tau)) d\tau$$

- Idea:
  - integrate linear part exactly (solution to homogeneous equation)
  - approximate the rest by exponential quadrature
- Left rule for the remainder integral leads to the exponential Euler method

$$\boldsymbol{u}(t_i + h_i) =: \boldsymbol{u}_{i+1} = e^{-h_i\boldsymbol{A}}\boldsymbol{u}_i + h_i\varphi_1(-h_i\boldsymbol{A})g(t_i, \boldsymbol{u}_i),$$

with

$$\varphi_1(z) = \frac{e^z - 1}{z}$$

- Starting point: variation-of-constants formula [Hochbruck, Ostermann, 2010]

$$\boldsymbol{u}(t) = e^{-t\boldsymbol{A}}\boldsymbol{u}_0 + \int_0^t e^{-(t-\tau)\boldsymbol{A}} g(\tau, \boldsymbol{u}(\tau)) d\tau$$

- Idea:
  - integrate linear part exactly (solution to homogeneous equation)
  - approximate the rest by exponential quadrature
- Left rule for the remainder integral leads to the exponential Euler method

$$\boldsymbol{u}(t_i + h_i) =: \boldsymbol{u}_{i+1} = e^{-h_i\boldsymbol{A}}\boldsymbol{u}_i + h_i\varphi_1(-h_i\boldsymbol{A})g(t_i, \boldsymbol{u}_i),$$

with

$$\varphi_1(z) = \frac{e^z - 1}{z}$$

- More sophisticated quadrature rules lead to more φ-functions satisfying the recursion

$$\varphi_0(z) = e^z, \quad \varphi_{k+1}(z) = \frac{\varphi_k(z) - \varphi_k(0)}{z} = \sum_{j=0}^{\infty} \frac{z^j}{(j+k+1)!},$$

  where $\varphi_k(0) = \frac{1}{k!}$.
- We use **explicit exponential Runge–Kutta (RK) methods**:

$$\begin{aligned} \boldsymbol{U}_{ij} &= \chi_j(-h_i\boldsymbol{A})\boldsymbol{u}_i + h_i \sum_{k=1}^{s} a_{jk}(-h_i\boldsymbol{A})\boldsymbol{G}_{ik}, \\ \boldsymbol{G}_{ik} &= g(t_i + c_k h_i, \boldsymbol{U}_{ik}), \\ \boldsymbol{u}_{i+1} &= \chi(-h_i\boldsymbol{A})\boldsymbol{u}_i + h_i \sum_{j=1}^{s} b_j(-h_i\boldsymbol{A})\boldsymbol{G}_{ij}. \end{aligned}$$

- Convergence order independent of problem stiffness.

We use

- SW2 (Strehmel & Weiner, stage $s = 2$, order $p = 2$) [Weiner, 2013]
- ETD3RK (Cox & Mathews, stage $s = 3$, order $p = 3$) [Cox, Mathews, 2002]
- `Krogstad4` (Krogstad, stage $s = 4$, order $p = 4$) [Krogstad, 2005]

$$
\begin{aligned}
\boldsymbol{U}_{i1} &= \boldsymbol{u}_i, \quad \boldsymbol{G}_{i1} = g(t_i, \boldsymbol{u}_i),\\
\boldsymbol{U}_{i2} &= \boldsymbol{u}_i + (h_i/2)\varphi_1(-(h_i/2)\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i), \quad \boldsymbol{G}_{i2} = g\left(t_i + (h_i/2), \boldsymbol{U}_{i2}\right),\\
\boldsymbol{U}_{i3} &= \boldsymbol{u}_i + h_i\left[(1/2)\varphi_1\left(-(h_i/2)\boldsymbol{A}\right)\left(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i\right) - \varphi_2(-(h_i/2)\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i)\right.\\
&\qquad \left. + \varphi_2(-(h_i/2)\boldsymbol{A})(\boldsymbol{G}_{i2} - \boldsymbol{A}\boldsymbol{u}_i)\right], \quad \boldsymbol{G}_{i3} = g\left(t_i + (h_i/2), \boldsymbol{U}_{i3}\right),\\
\boldsymbol{U}_{i4} &= \boldsymbol{u}_i + h_i\left[\varphi_1(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i) - 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i)\right.\\
&\qquad \left. + 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i3} - \boldsymbol{A}\boldsymbol{u}_i)\right], \quad \boldsymbol{G}_{i4} = g(t_i + h_i, \boldsymbol{U}_{i4}),\\
\boldsymbol{u}_{i+1} &= \boldsymbol{u}_i + h_i\left[\varphi_1(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i) - 3\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i)\right.\\
&\qquad + 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i)\\
&\qquad + 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i2} - \boldsymbol{A}\boldsymbol{u}_i) - 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i2} - \boldsymbol{A}\boldsymbol{u}_i)\\
&\qquad + 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i3} - \boldsymbol{A}\boldsymbol{u}_i) - 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i3} - \boldsymbol{A}\boldsymbol{u}_i)\\
&\qquad \left. - \varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i4} - \boldsymbol{A}\boldsymbol{u}_i) + 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i4} - \boldsymbol{A}\boldsymbol{u}_i)\right].
\end{aligned}
$$

- Naively implemented, computation of linear combinations of φ-functions times a vector very expensive

We use

- SW2 (Strehmel & Weiner, stage $s = 2$, order $p = 2$) [Weiner, 2013]
- ETD3RK (Cox & Mathews, stage $s = 3$, order $p = 3$) [Cox, Mathews, 2002]
- Krogstad4 (Krogstad, stage $s = 4$, order $p = 4$) [Krogstad, 2005]

$$
\begin{aligned}
\boldsymbol{U}_{i1} &= \boldsymbol{u}_i, \quad \boldsymbol{G}_{i1} = g(t_i, \boldsymbol{u}_i), \\
\boldsymbol{U}_{i2} &= \boldsymbol{u}_i + (h_i/2)\varphi_1(-(h_i/2)\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i), \quad \boldsymbol{G}_{i2} = g\left(t_i + (h_i/2), \boldsymbol{U}_{i2}\right), \\
\boldsymbol{U}_{i3} &= \boldsymbol{u}_i + h_i\left[(1/2)\varphi_1\left(-(h_i/2)\boldsymbol{A}\right)\left(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i\right) - \varphi_2(-(h_i/2)\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i)\right. \\
&\qquad \left. + \varphi_2(-(h_i/2)\boldsymbol{A})(\boldsymbol{G}_{i2} - \boldsymbol{A}\boldsymbol{u}_i)\right], \quad \boldsymbol{G}_{i3} = g\left(t_i + (h_i/2), \boldsymbol{U}_{i3}\right), \\
\boldsymbol{U}_{i4} &= \boldsymbol{u}_i + h_i\left[\varphi_1(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i) - 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i)\right. \\
&\qquad \left. + 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i3} - \boldsymbol{A}\boldsymbol{u}_i)\right], \quad \boldsymbol{G}_{i4} = g(t_i + h_i, \boldsymbol{U}_{i4}), \\
\boldsymbol{u}_{i+1} &= \boldsymbol{u}_i + h_i\left[\varphi_1(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i) - 3\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i)\right. \\
&\qquad + 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i1} - \boldsymbol{A}\boldsymbol{u}_i) \\
&\qquad + 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i2} - \boldsymbol{A}\boldsymbol{u}_i) - 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i2} - \boldsymbol{A}\boldsymbol{u}_i) \\
&\qquad + 2\varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i3} - \boldsymbol{A}\boldsymbol{u}_i) - 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i3} - \boldsymbol{A}\boldsymbol{u}_i) \\
&\qquad \left. - \varphi_2(-h_i\boldsymbol{A})(\boldsymbol{G}_{i4} - \boldsymbol{A}\boldsymbol{u}_i) + 4\varphi_3(-h_i\boldsymbol{A})(\boldsymbol{G}_{i4} - \boldsymbol{A}\boldsymbol{u}_i)\right].
\end{aligned}
$$

- Naively implemented, computation of linear combinations of φ-functions times a vector very expensive

Exponential integrators

Efficient implementation

- Saad found φ_1 in the matrix exp. of an extended Hessenberg matrix

Theorem 1 (Saad, 1992)

For $\widetilde{\boldsymbol{H}}_{m+1} = \begin{pmatrix} \boldsymbol{H}_m & \boldsymbol{c} \\ \boldsymbol{0} & 0 \end{pmatrix}$ *with* $\boldsymbol{c} \in \mathbb{R}^m$ *we have* $e^{\widetilde{\boldsymbol{H}}_{m+1}} = \begin{pmatrix} e^{\boldsymbol{H}_m} & \varphi_1(\boldsymbol{H}_m)\boldsymbol{c} \\ \boldsymbol{0} & 1 \end{pmatrix}$.

- Sidje extended this to the first p φ-functions

Theorem 2 (Sidje, 1998)

For $\widetilde{\boldsymbol{H}}_{m+p} = \begin{pmatrix} \boldsymbol{H}_m & \boldsymbol{c} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_{p-1} \\ \boldsymbol{0} & 0 & \boldsymbol{0} \end{pmatrix}$ *with* $\boldsymbol{c} \in \mathbb{R}^m$ *we have*

$$e^{h_i \widetilde{\boldsymbol{H}}_{m+p}} = \begin{pmatrix} e^{h_i \boldsymbol{H}_m} & h_i \varphi_1(h_i \boldsymbol{H}_m)\boldsymbol{c} & h_i^2 \varphi_2(h_i \boldsymbol{H}_m)\boldsymbol{c} & \cdots & h_i^p \varphi_p(h_i \boldsymbol{H}_m)\boldsymbol{c} \\ & 1 & \frac{h_i}{1!} & \cdots & \frac{h_i^{p-1}}{(p-1)!} \\ & & 1 & \ddots & \vdots \\ & & & \ddots & \frac{h_i}{1!} \\ 0 & & & & 1 \end{pmatrix}.$$

- Most general version by Al-Mohy, Higham in our notation

Theorem 3 (Al-Mohy, Higham, 2011)

Let $\widetilde{\boldsymbol{A}} = \begin{pmatrix} -\boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{0} & \boldsymbol{J}_p \end{pmatrix} \in \mathbb{C}^{(n+p)\times(n+p)}$*, where* $\boldsymbol{C} = [\boldsymbol{c}_p, \ldots, \boldsymbol{c}_1] \in \mathbb{C}^{n\times p}$ *and* $\boldsymbol{J}_p \in \mathbb{C}^{p\times p}$ *a Jordan block to the eigenvalue* 0*. Furthermore, we define the matrix exponential* $\boldsymbol{X} = e^{h_i \widetilde{\boldsymbol{A}}}$ *as well as the vector* $\widetilde{\boldsymbol{c}} := \begin{pmatrix} \boldsymbol{c}_0 \\ \boldsymbol{e}_p \end{pmatrix} \in \mathbb{C}^{n+p}$*. Then, we have* $\boldsymbol{X}(1:n, n+p) = \sum_{k=1}^{p} h_i^k \varphi_k(-h_i \boldsymbol{A})\boldsymbol{c}_k$ *and*

$$\boldsymbol{X}\widetilde{\boldsymbol{c}} = e^{h_i \widetilde{\boldsymbol{A}}}\widetilde{\boldsymbol{c}} = \begin{pmatrix} \sum_{k=0}^{p} h_i^k \varphi_k(-h_i \boldsymbol{A})\boldsymbol{c}_k \\ e^{\boldsymbol{J}_p}\boldsymbol{e}_p \end{pmatrix} := \widetilde{\boldsymbol{b}} \in \mathbb{C}^{n+p}.$$

- We are interested in $\widetilde{b}(1:n)$, the rest can be discarded

Remark

The spectrum of $\widetilde{\boldsymbol{A}}$ *is the union of the spectrum of* $-\boldsymbol{A}$ *with the eigenvalue* 0 *with multiplicity* p *independently of the matrix* $\boldsymbol{C}$*.*

Use above results for efficient implementations of exponential integrators!

1. Niesen, Wright use Theorem 2 for `phipm` [Niesen, Wright, 2012]
2. Gaudreault, Rainwater, Tokman use Theorem 3 for `KIOPS` [Gaudreault, Rainwater, Tokman, 2018]

Common idea:

- Group φ-function terms in exponential integrators and approximate

$$\varphi_0(-h_i \boldsymbol{A})\boldsymbol{c}_0 + h_i\varphi_1(-h_i \boldsymbol{A})\boldsymbol{c}_1 + h_i^2\varphi_2(-h_i \boldsymbol{A})\boldsymbol{c}_2 + \cdots + h_i^p\varphi_p(-h_i \boldsymbol{A})\boldsymbol{c}_p$$

  computing only one matrix exponential via polynomial Krylov methods [Saad, 1992], [Hochbruck, Lubich, 1997]
- Adaptivity based on error estimate:
  - number of Krylov subspace iterations
  - number of sub-steps in $[t_i, t_{i+1}]$

Use above results for efficient implementations of exponential integrators!

1. Niesen, Wright use Theorem 2 for `phipm` [Niesen, Wright, 2012]
2. Gaudreault, Rainwater, Tokman use Theorem 3 for `KIOPS` [Gaudreault, Rainwater, Tokman, 2018]

Common idea:

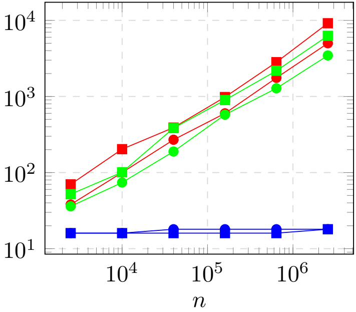
- Group φ-function terms in exponential integrators and approximate

$$\varphi_0(-h_i \boldsymbol{A})\boldsymbol{c}_0 + h_i \varphi_1(-h_i \boldsymbol{A})\boldsymbol{c}_1 + h_i^2 \varphi_2(-h_i \boldsymbol{A})\boldsymbol{c}_2 + \cdots + h_i^p \varphi_p(-h_i \boldsymbol{A})\boldsymbol{c}_p$$

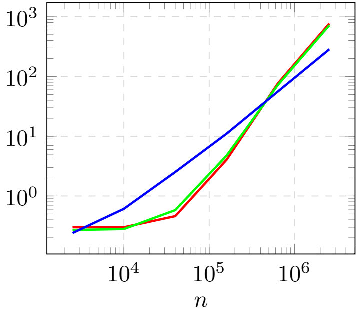
  computing only one matrix exponential via polynomial Krylov methods [Saad, 1992], [Hochbruck, Lubich, 1997]

- Adaptivity based on error estimate:
  - number of Krylov subspace iterations
  - number of sub-steps in $[t_i, t_{i+1}]$

phipm and KIOPS are great! But there is one problem:



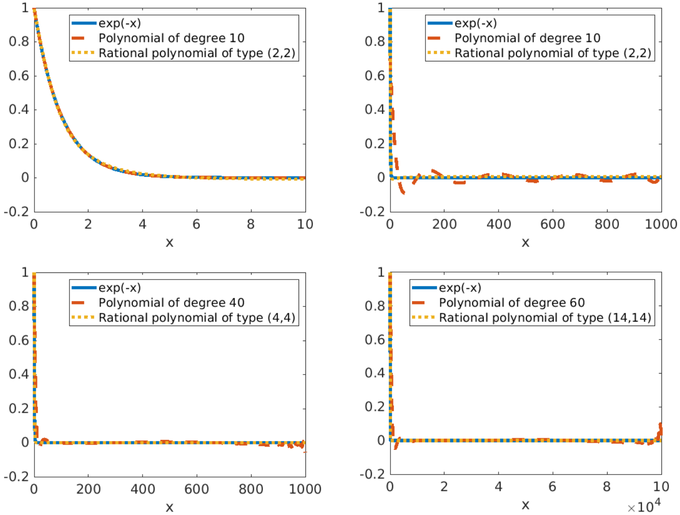
(a) Krylov iteration numbers



(b) Runtime in seconds

Figure: Scaling of 2D Allen–Cahn example

Why this problem? For $\boldsymbol{A} = \boldsymbol{A}^T = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^T \in \mathbb{R}^{n \times n}$, we have $e^{-h_i \boldsymbol{A}} = \boldsymbol{\Phi} e^{-h_i \boldsymbol{\Lambda}} \boldsymbol{\Phi}^T = \sum_{j=1}^{n} e^{-h_i \lambda_j} \phi_j \phi_j^T$.



Rational Krylov subspace methods

- phipm and KIOPS: approximate $f(\widetilde{\boldsymbol{A}})\widetilde{\boldsymbol{c}}$ by projecting the matrix $\widetilde{\boldsymbol{A}}$ onto the polynomial Krylov subspace

$$\mathcal{K}_m(\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{c}}) = \text{span}\{\widetilde{\boldsymbol{c}}, \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{c}}, \ldots, \widetilde{\boldsymbol{A}}^{m-1}\widetilde{\boldsymbol{c}}\}$$

and then use a rational Padé approximation to compute $f(\boldsymbol{H}_m)$.

- Alternative: project $\widetilde{\boldsymbol{A}}$ onto **Rational Krylov (RK) subspace** [Güttel, 2013]

$$\mathcal{Q}_m(\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{c}}) = q_{m-1}(\widetilde{\boldsymbol{A}})^{-1}\text{span}\{\widetilde{\boldsymbol{c}}, \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{c}}, \ldots, \widetilde{\boldsymbol{A}}^{m-1}\widetilde{\boldsymbol{c}}\}$$

with $q_{m-1}(\widetilde{\boldsymbol{A}})$ a matrix polynomial of degree $m-1$, which we assume to be factored as

$$q_{m-1}(z) = \prod_{j=1}^{m-1}(1 - z/\xi_j).$$

  - The $\xi_j \in \mathbb{C} \cup \{\infty\}, j = 1, \ldots, m-1$ are the poles of q_{m-1}
  - We require $0 \neq \xi_j \notin \sigma(\widetilde{\boldsymbol{A}})$ to ensure the invertibility of $q_{m-1}(\widetilde{\boldsymbol{A}})$
  - Special cases:
    - $\xi_1 = \cdots = \xi_{m-1} = \xi$: shift & invert Krylov methods
    - $\xi_1 = \cdots = \xi_{m-1} = \infty$: polynomial Krylov methods

- `phipm` and `KIOPS`: approximate $f(\widetilde{\boldsymbol{A}})\widetilde{\boldsymbol{c}}$ by projecting the matrix $\widetilde{\boldsymbol{A}}$ onto the polynomial Krylov subspace

$$\mathcal{K}_m(\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{c}}) = \text{span}\{\widetilde{\boldsymbol{c}}, \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{c}}, \dots, \widetilde{\boldsymbol{A}}^{m-1}\widetilde{\boldsymbol{c}}\}$$

  and then use a rational Padé approximation to compute $f(\boldsymbol{H}_m)$.
- Alternative: project $\widetilde{\boldsymbol{A}}$ onto **Rational Krylov (RK) subspace** [Güttel, 2013]

$$\mathcal{Q}_m(\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{c}}) = q_{m-1}(\widetilde{\boldsymbol{A}})^{-1}\text{span}\{\widetilde{\boldsymbol{c}}, \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{c}}, \dots, \widetilde{\boldsymbol{A}}^{m-1}\widetilde{\boldsymbol{c}}\}$$

  with $q_{m-1}(\widetilde{\boldsymbol{A}})$ a matrix polynomial of degree $m-1$, which we assume to be factored as

$$q_{m-1}(z) = \prod_{j=1}^{m-1}(1 - z/\xi_j).$$

  - The $\xi_j \in \mathbb{C} \cup \{\infty\}, j = 1, \dots, m-1$ are the poles of q_{m-1}
  - We require $0 \neq \xi_j \notin \sigma(\widetilde{\boldsymbol{A}})$ to ensure the invertibility of $q_{m-1}(\widetilde{\boldsymbol{A}})$
  - Special cases:
    - $\xi_1 = \cdots = \xi_{m-1} = \xi$: shift & invert Krylov methods
    - $\xi_1 = \cdots = \xi_{m-1} = \infty$: polynomial Krylov methods

- Computation of a basis $\boldsymbol{V}_{m+1}$ of the rational Krylov subspace: Ruhe's rational Arnoldi algorithm [Ruhe, 1994]
  - Set $\boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|$
  - Replace $\boldsymbol{x}_{j+1} = \widetilde{\boldsymbol{A}}\boldsymbol{v}_j$ by $\boldsymbol{x}_{j+1} = (\boldsymbol{I} - \widetilde{\boldsymbol{A}}/\xi_j)^{-1}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{v}}_j$
  - Orthonormalize $\boldsymbol{x}_{j+1}$ against $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_j$ to obtain $\boldsymbol{v}_{j+1}$
- More expensive per iteration (linear system solve), but can pay off due to superior approximation quality
- Yields the projection

$$\underline{\boldsymbol{H}_m} = \boldsymbol{V}_{m+1}^T \widetilde{\boldsymbol{A}} \boldsymbol{V}_{m+1} \underline{\boldsymbol{K}_m} \in \mathbb{C}^{(m+1)\times m}$$

with

$$\underline{\boldsymbol{H}_m} = \begin{pmatrix} \boldsymbol{H}_m \\ h_{m+1,m}\boldsymbol{e}_m^* \end{pmatrix}, \quad \underline{\boldsymbol{K}_m} = \begin{pmatrix} \boldsymbol{I}_m + \boldsymbol{H}_m \boldsymbol{D}_k \\ h_{m+1,m}\xi_m^{-1}\boldsymbol{e}_m^* \end{pmatrix},$$

where $\boldsymbol{D}_m = \text{diag}(\xi_1^{-1}, \ldots, \xi_m^{-1})$.
- For $\xi_m = \infty$, this leads to

$$f(\widetilde{\boldsymbol{A}})\widetilde{\boldsymbol{c}} \approx \|\widetilde{\boldsymbol{c}}\|_2 \boldsymbol{V}_m f(\boldsymbol{H}_m \boldsymbol{K}_m^{-1}) e_1.$$

- Computation of a basis $\boldsymbol{V}_{m+1}$ of the rational Krylov subspace: Ruhe's rational Arnoldi algorithm [Ruhe, 1994]
  - Set $\boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|$
  - Replace $\boldsymbol{x}_{j+1} = \widetilde{\boldsymbol{A}}\boldsymbol{v}_j$ by $\boldsymbol{x}_{j+1} = (\boldsymbol{I} - \widetilde{\boldsymbol{A}}/\xi_j)^{-1}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{v}}_j$
  - Orthonormalize $\boldsymbol{x}_{j+1}$ against $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_j$ to obtain $\boldsymbol{v}_{j+1}$
- More expensive per iteration (linear system solve), but can pay off due to superior approximation quality
- Yields the projection

$$\underline{\boldsymbol{H}_m} = \boldsymbol{V}_{m+1}^T \widetilde{\boldsymbol{A}} \boldsymbol{V}_{m+1} \underline{\boldsymbol{K}_m} \in \mathbb{C}^{(m+1)\times m}$$

with

$$\underline{\boldsymbol{H}_m} = \begin{pmatrix} \boldsymbol{H}_m \\ h_{m+1,m}\boldsymbol{e}_m^* \end{pmatrix}, \quad \underline{\boldsymbol{K}_m} = \begin{pmatrix} \boldsymbol{I}_m + \boldsymbol{H}_m \boldsymbol{D}_k \\ h_{m+1,m}\xi_m^{-1}\boldsymbol{e}_m^* \end{pmatrix},$$

where $\boldsymbol{D}_m = \operatorname{diag}(\xi_1^{-1}, \ldots, \xi_m^{-1})$.

- For $\xi_m = \infty$, this leads to

$$f(\widetilde{\boldsymbol{A}})\widetilde{\boldsymbol{c}} \approx \|\widetilde{\boldsymbol{c}}\|_2 \boldsymbol{V}_m f(\boldsymbol{H}_m \boldsymbol{K}_m^{-1}) e_1.$$

- Computation of a basis $\boldsymbol{V}_{m+1}$ of the rational Krylov subspace: Ruhe's rational Arnoldi algorithm [Ruhe, 1994]
  - Set $\boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|$
  - Replace $\boldsymbol{x}_{j+1} = \widetilde{\boldsymbol{A}}\boldsymbol{v}_j$ by $\boldsymbol{x}_{j+1} = (\boldsymbol{I} - \widetilde{\boldsymbol{A}}/\xi_j)^{-1}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{v}}_j$
  - Orthonormalize $\boldsymbol{x}_{j+1}$ against $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_j$ to obtain $\boldsymbol{v}_{j+1}$
- More expensive per iteration (linear system solve), but can pay off due to superior approximation quality
- Yields the projection

$$\underline{\boldsymbol{H}_m} = \boldsymbol{V}_{m+1}^T \widetilde{\boldsymbol{A}} \boldsymbol{V}_{m+1} \underline{\boldsymbol{K}_m} \in \mathbb{C}^{(m+1)\times m}$$

with

$$\underline{\boldsymbol{H}_m} = \begin{pmatrix} \boldsymbol{H}_m \\ h_{m+1,m}\boldsymbol{e}_m^* \end{pmatrix}, \quad \underline{\boldsymbol{K}_m} = \begin{pmatrix} \boldsymbol{I}_m + \boldsymbol{H}_m \boldsymbol{D}_k \\ h_{m+1,m}\xi_m^{-1}\boldsymbol{e}_m^* \end{pmatrix},$$

where $\boldsymbol{D}_m = \text{diag}(\xi_1^{-1}, \ldots, \xi_m^{-1})$.
- For $\xi_m = \infty$, this leads to

$$f(\widetilde{\boldsymbol{A}})\widetilde{\boldsymbol{c}} \approx \|\widetilde{\boldsymbol{c}}\|_2 \boldsymbol{V}_m f(\boldsymbol{H}_m \boldsymbol{K}_m^{-1})\boldsymbol{e}_1.$$

$$(RK)^2\text{EXPINT} = \texttt{KIOPS} - \mathcal{K}_m(\widetilde{\boldsymbol{A}}, \boldsymbol{b}) + \mathcal{Q}_m(\widetilde{\boldsymbol{A}}, \boldsymbol{b})$$

Implementation: rktoolbox [Berljafa, Elsworth, Güttel, 2014]

Questions:

1. What poles to use?
2. How to solve the linear systems efficiently?
3. When to stop?

$$(RK)^2\text{EXPINT} = \texttt{KIOPS} - \mathcal{K}_m(\widetilde{\boldsymbol{A}}, \boldsymbol{b}) + \mathcal{Q}_m(\widetilde{\boldsymbol{A}}, \boldsymbol{b})$$

Implementation: rktoolbox [Berljafa, Elsworth, Güttel, 2014]

Questions:

1. What poles to use?
2. How to solve the linear systems efficiently?
3. When to stop?

Rational Krylov subspace methods

Pole selection

- Rational best approximation to e^{-x} on $[0, \infty)$: [Cody, Meinardus, Varga, 1969], [Carpenter, Ruttan, Varga, 1984], [Gallopoulos, Saad, 1992]
  - Find
    $$r_{d,d}(x) = \frac{p_d(x)}{q_d(x)}, \quad p_d, q_d \in \mathbb{P}_d,$$
    that minimizes
    $$\sup_{0 \leq x < \infty} \left| r_{d,d}(x) - e^{-x} \right|.$$
  - Coefficients of optimal p_d and q_d tabulated up to $d = 30$ (at least)
  - Take complex conjugated roots of q_d as poles
  - Roots of q_d have positive and negative real and imaginary parts
- Restriction to real poles leads to a single repeated real pole and shift & invert Krylov methods [Moret, Novati, 2004] [Van Den Eshof, Hochbruck, 2006]
- RKFIT poles of q_d for rational polynomials of type $(m + k, m)$ and finite interval $[0, \lambda_{\max}]$ [Berljafa, Güttel, 2015] [Berljafa, Güttel, 2017]
  - Poles can be restricted to one complex half plane
  - Implemented in the rktoolbox [Berljafa, Elsworth, Güttel, 2014]

- Rational best approximation to e^{-x} on $[0, \infty)$: [Cody, Meinardus, Varga, 1969], [Carpenter, Ruttan, Varga, 1984], [Gallopoulos, Saad, 1992]
  - Find
$$r_{d,d}(x) = \frac{p_d(x)}{q_d(x)}, \quad p_d, q_d \in \mathbb{P}_d,$$
that minimizes
$$\sup_{0 \leq x < \infty} |r_{d,d}(x) - e^{-x}|.$$
  - Coefficients of optimal p_d and q_d tabulated up to $d = 30$ (at least)
  - Take complex conjugated roots of q_d as poles
  - Roots of q_d have positive and negative real and imaginary parts
- Restriction to real poles leads to a single repeated real pole and shift & invert Krylov methods [Moret, Novati, 2004] [Van Den Eshof, Hochbruck, 2006]
- RKFIT poles of q_d for rational polynomials of type $(m+k, m)$ and finite interval $[0, \lambda_{\max}]$ [Berljafa, Güttel, 2015] [Berljafa, Güttel, 2017]
  - Poles can be restricted to one complex half plane
  - Implemented in the rktoolbox [Berljafa, Elsworth, Güttel, 2014]

- Rational best approximation to e^{-x} on $[0, \infty)$: [Cody, Meinardus, Varga, 1969], [Carpenter, Ruttan, Varga, 1984], [Gallopoulos, Saad, 1992]
  - Find
    $$r_{d,d}(x) = \frac{p_d(x)}{q_d(x)}, \quad p_d, q_d \in \mathbb{P}_d,$$
    that minimizes
    $$\sup_{0 \le x < \infty} |r_{d,d}(x) - e^{-x}|.$$
  - Coefficients of optimal p_d and q_d tabulated up to $d = 30$ (at least)
  - Take complex conjugated roots of q_d as poles
  - Roots of q_d have positive and negative real and imaginary parts
- Restriction to real poles leads to a single repeated real pole and shift & invert Krylov methods [Moret, Novati, 2004] [Van Den Eshof, Hochbruck, 2006]
- RKFIT poles of q_d for rational polynomials of type $(m + k, m)$ and finite interval $[0, \lambda_{\max}]$ [Berljafa, Güttel, 2015] [Berljafa, Güttel, 2017]
  - Poles can be restricted to one complex half plane
  - Implemented in the rktoolbox [Berljafa, Elsworth, Güttel, 2014]

Rational Krylov subspace methods

Linear system solves

- Linear system solves in rational Arnoldi update (for $\boldsymbol{b}_j := \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{v}}_j$),

$$\boldsymbol{x}_{j+1} = (\boldsymbol{I} - \widetilde{\boldsymbol{A}}/\xi_j)^{-1}\boldsymbol{b}_j \Leftrightarrow (\boldsymbol{I} - \widetilde{\boldsymbol{A}}/\xi_j)\boldsymbol{x}_{j+1} = \boldsymbol{b}_j \Leftrightarrow (\xi_j \boldsymbol{I} - \widetilde{\boldsymbol{A}})\boldsymbol{x}_{j+1} = \xi_j \boldsymbol{b}_j.$$

- By the definition of $\widetilde{\boldsymbol{A}}$,

$$(\xi_j \boldsymbol{I}_{n+p} - \widetilde{\boldsymbol{A}})\boldsymbol{x}_{j+1} = \begin{bmatrix} \xi_j \boldsymbol{I}_n + \boldsymbol{A} & -\boldsymbol{C} \\ \boldsymbol{0} & \xi_j \boldsymbol{I}_p - \boldsymbol{J}_p \end{bmatrix} \begin{bmatrix} [\boldsymbol{x}_{j+1}]_n \\ [\boldsymbol{x}_{j+1}]_p \end{bmatrix} = \xi_j \begin{bmatrix} [\boldsymbol{b}_j]_n \\ [\boldsymbol{b}_j]_p \end{bmatrix},$$

- As $p \ll n$, we efficiently solve for $[\boldsymbol{x}_{j+1}]_p$ and backsubstitute to obtain

$$(\xi_j \boldsymbol{I}_n + \boldsymbol{A})[\boldsymbol{x}_{j+1}]_n = \xi_j [\boldsymbol{b}_j]_n + \boldsymbol{C}[\boldsymbol{x}_{j+1}]_p.$$

- Since $\boldsymbol{A}$ is pos. semi-def., it helps a lot when ξ_j have positive real parts

2 strategies:

- Direct solvers (LU/Cholesky decomposition)
  - Compute decomposition for each $(\xi_j \boldsymbol{I}_n + \boldsymbol{A})$ (and potentially $(\xi_j \boldsymbol{I}_n + c_j h_i \boldsymbol{A})$), depending on the integrator) upfront
  - Cheaply solve linear systems with factors
  - Very efficient for many time steps
  - Problem size limited by memory requirement
  - Use optimized permutations to avoid fill-in
  - Implementation: Pardiso 6.0 [Petra, Schenk, Anitescu, 2014] [Petra, Schenk, Lubin, Gärtner, 2014]
- Preconditioned iterative solvers (CG/MINRES/GMRES)
  - Unpreconditioned, they suffer from the very issue of increasing polynomial Krylov subspace sizes we try to avoid
  - Use preconditioner $\boldsymbol{P} \approx \boldsymbol{A}$ to solve $\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{P}^{-1}\boldsymbol{b} \Leftrightarrow \boldsymbol{P}^{-1}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = 0$
  - Leads to approximately constant iteration numbers and linear scaling w.r.t. the problem size
  - Algebraic Multigrid (AMG) well-suited if Re$(\xi_j) > 0$
  - Implementation: Aggregation-based multigrid package (AGMG) 3.3.5 [Notay, 2010] [Notay, 2012] [Napov, Notay, 2012]

2 strategies:

- Direct solvers (LU/Cholesky decomposition)
  - Compute decomposition for each $(\xi_j \boldsymbol{I}_n + \boldsymbol{A})$ (and potentially $(\xi_j \boldsymbol{I}_n + c_j h_i \boldsymbol{A})$), depending on the integrator) upfront
  - Cheaply solve linear systems with factors
  - Very efficient for many time steps
  - Problem size limited by memory requirement
  - Use optimized permutations to avoid fill-in
  - Implementation: Pardiso 6.0 [Petra, Schenk, Anitescu, 2014] [Petra, Schenk, Lubin, Gärtner, 2014]
- Preconditioned iterative solvers (CG/MINRES/GMRES)
  - Unpreconditioned, they suffer from the very issue of increasing polynomial Krylov subspace sizes we try to avoid
  - Use preconditioner $\boldsymbol{P} \approx \boldsymbol{A}$ to solve $\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{P}^{-1}\boldsymbol{b} \Leftrightarrow \boldsymbol{P}^{-1}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = 0$
  - Leads to approximately constant iteration numbers and linear scaling w.r.t. the problem size
  - Algebraic Multigrid (AMG) well-suited if Re$(\xi_j) > 0$
  - Implementation: Aggregation-based multigrid package (AGMG) 3.3.5 [Notay, 2010] [Notay, 2012] [Napov, Notay, 2012]

Rational Krylov subspace methods

A-posteriori error estimate

- Fun fact: initial Theorem by Saad on computing $\varphi_1(\widetilde{\boldsymbol{A}})\widetilde{\boldsymbol{c}}$ had nothing to do with exponential integrators
- Instead:

Theorem 4 (Saad, 1992)

The error produced by the [polynomial] Arnoldi or Lanczos approximation satisfies the following expansion:

$$e^{\widetilde{\boldsymbol{A}}}\widetilde{\boldsymbol{c}} - \boldsymbol{V}_m e^{\boldsymbol{H}_m}\boldsymbol{e}_1 = h_{m+1,m}\sum_{k=1}^{\infty} \boldsymbol{e}_m^T \varphi_k(\boldsymbol{H}_m)\boldsymbol{e}_1 \widetilde{\boldsymbol{A}}^{k-1}\boldsymbol{v}_{m+1},$$

where $\|\widetilde{\boldsymbol{c}}\|_2 = 1$.

Truncation of the sum leads to the practical error estimate

$$\|e^{\widetilde{\boldsymbol{A}}}\widetilde{\boldsymbol{c}} - \boldsymbol{V}_m e^{\boldsymbol{H}_m}\boldsymbol{e}_1\|_2 \approx h_{m+1,m}|\boldsymbol{e}_m^T\varphi_1(\boldsymbol{H}_m)\boldsymbol{e}_1|.$$

- Yields stopping criterion in `KIOPS`.

Rational Krylov relation for $\xi_m = \infty$:

$$\widetilde{\boldsymbol{A}}\boldsymbol{V}_m\boldsymbol{K}_m = \boldsymbol{V}_m\boldsymbol{H}_m + h_{m+1,m}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^* \in \mathbb{C}^{n\times m}.$$

Theorem 5 (B., Stoll, 2024)

Let $\xi_m = \infty$. Then the approximation error of the rational Krylov approximation $\|\widetilde{\boldsymbol{c}}\|_2\boldsymbol{V}_m e^{h_i\boldsymbol{H}_m\boldsymbol{K}_m^{-1}}\boldsymbol{e}_1$ to $e^{h_i\widetilde{\boldsymbol{A}}}\widetilde{\boldsymbol{c}}$ reads

$$e^{h_i\widetilde{\boldsymbol{A}}}\widetilde{\boldsymbol{c}} - \|\widetilde{\boldsymbol{c}}\|_2\boldsymbol{V}_m e^{h_i\boldsymbol{H}_m\boldsymbol{K}_m^{-1}}\boldsymbol{e}_1 = h_i\|\widetilde{\boldsymbol{c}}\|_2 h_{m+1,m}\sum_{k=1}^{\infty}\boldsymbol{e}_m^*\boldsymbol{K}_m^{-1}\varphi_k(h_i\boldsymbol{H}_m\boldsymbol{K}_m^{-1})\boldsymbol{e}_1(h_i\widetilde{\boldsymbol{A}})^{k-1}\boldsymbol{v}_{m+1}.$$

- Truncation of the sum leads to the practical error estimate

$$\|e^{h_i \widetilde{\boldsymbol{A}}} \widetilde{\boldsymbol{c}} - \|\widetilde{\boldsymbol{c}}\|_2 \boldsymbol{V}_m e^{h_i \boldsymbol{H}_m \boldsymbol{K}_m^{-1}} \boldsymbol{e}_1\|_2 \approx h_i \|\widetilde{\boldsymbol{c}}\|_2 h_{m+1,m} \left| \boldsymbol{e}_m^* \boldsymbol{K}_m^{-1} \varphi_1(h_i \boldsymbol{H}_m \boldsymbol{K}_m^{-1}) \boldsymbol{e}_1 \right|$$

- Being experts on the approximation of φ-functions, we know that for

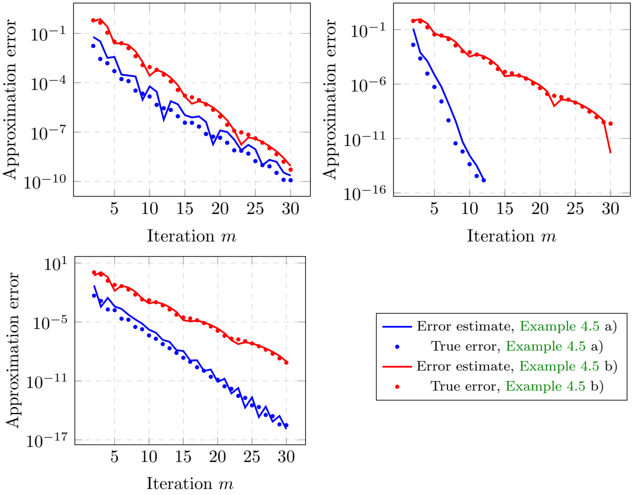
$$\boldsymbol{M}_{m+1} := \begin{bmatrix} \boldsymbol{H}_m \boldsymbol{K}_m^{-1} & \boldsymbol{e}_1 \\ \boldsymbol{0}^T & 0 \end{bmatrix} \in \mathbb{C}^{(m+1)\times(m+1)},$$

we get

$$e^{h_i \boldsymbol{M}_{m+1}} = \begin{bmatrix} e^{h_i \boldsymbol{H}_m \boldsymbol{K}_m^{-1}} & h_i \varphi_1(h_i \boldsymbol{H}_m \boldsymbol{K}_m^{-1}) \boldsymbol{e}_1 \\ \boldsymbol{0}^T & 1 \end{bmatrix}.$$

- Yields stopping criterion for $(RK)^2$EXPINT.

And it works:


Approximation error
Iteration m
10^{-1}
10^{-4}
10^{-7}
10^{-10}
10^{-6}
10^{-11}
10^{-16}
10^{1}
10^{-5}
10^{-17}
5 10 15 20 25 30
Error estimate, Example 4.5 a)
True error, Example 4.5 a)
Error estimate, Example 4.5 b)
True error, Example 4.5 b)

Algorithm

Algorithm

Input: $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, Discrete linear differential operator.
$g : [0, T] \times \mathbb{R}^n \to \mathbb{R}^n$, Semi-linear function.
$\boldsymbol{u_0} \in \mathbb{R}^n$, Initial conditions.
$[0, T] \subset \mathbb{R}_{\geq 0}$, Time interval.

Parameters: $h_i \in \mathbb{R}_{>0}$; $\texttt{tol} \in \mathbb{R}_{>0}$; $\texttt{m\_min}, \texttt{m\_max} \in \mathbb{N}$; $\xi_j \in \mathbb{C}, j = 1, \ldots, \texttt{m\_max}$

Subroutines: `exp_rk_int`, `exptAb_routine`, `linear_system_solver`

if `linear_system_solver` == *direct* **then**
    Compute decompositions of $(\xi_j \boldsymbol{I}_n + \boldsymbol{A})$ for $j = 1, \ldots, \texttt{m\_max}$
end if
function `exp_rk_int` *% solve* (2.1)
for every time step **do**
    for each linear combination of φ-functions **do**
        Assemble $\widetilde{\boldsymbol{A}}$ and $\widetilde{\boldsymbol{c}}$

Algorithm

function `exptAb_routine` *% approximate* $e^{h_i \widetilde{\boldsymbol{A}}} \widetilde{\boldsymbol{c}}$
  while (4.12) $<$ `tol` **do**
    Compute continuation vector $\widetilde{\boldsymbol{v}}_j$ ($\widetilde{\boldsymbol{v}}_j = \boldsymbol{v}_j$ if not `rk2expint`)
    Compute $\boldsymbol{b}_j = \widetilde{\boldsymbol{A}} \widetilde{\boldsymbol{v}}_j$
    if `exptAb_routine` $==$ `rk2expint` $\&\&$ $j <$ `m_max` **then**
      if `linear_system_solver` $==$ *direct* **then**
        Solve (4.4) with back-subst. and the decomposition of $(\xi_j \boldsymbol{I}_n + \boldsymbol{A})$
      else if `linear_system_solver` $==$ *iterative* **then**
        Setup AGMG hierarchy for $(\xi_j \boldsymbol{I}_n + \boldsymbol{A})$
        Solve (4.4) with back-subst. and iterative AGMG solver
      end if
    end if
    Extend Krylov decomposition, i.e, $\boldsymbol{V}_m$, $\boldsymbol{H}_m$ (and $\boldsymbol{K}_m$ if `rk2expint`)
    Compute $\|\widetilde{\boldsymbol{c}}\|_2 \boldsymbol{V}_m e^{h_i \boldsymbol{H}_m \boldsymbol{K}_m^{-1}} \boldsymbol{e}_1$
  end while
end `exptAb_routine`
end for
Update solution $\boldsymbol{u}$ for current time step according to (3.2)–(3.4)
end for
end `exp_rk_int`

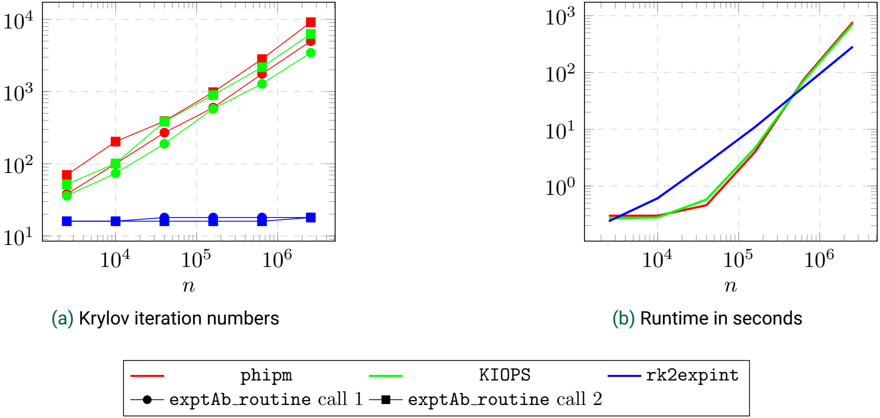
Output: $\boldsymbol{u} \in \mathbb{R}^{n \times n_t}$ Trajectory of the solution of (2.1) along the n_t time steps.

Numerical experiments

Numerical experiments

2D Allen–Cahn

$$\frac{\partial u}{\partial t} = \epsilon^2 \Delta u + u - u^3, \quad \epsilon \in \mathbb{R}.$$



(a) Krylov iteration numbers

(b) Runtime in seconds

Figure: Scaling of 2D Allen–Cahn example.

Numerical experiments

2D Gierer–Meinhardt

Gierer–Meinhardt equations

$$\frac{\partial a}{\partial t} = D_a \Delta a + p\frac{a^2}{h} - \mu a,$$
$$\frac{\partial h}{\partial t} = D_h \Delta h + p' a^2 - \nu h, \quad D_a, D_h, p, p', \mu, \nu \in \mathbb{R}.$$

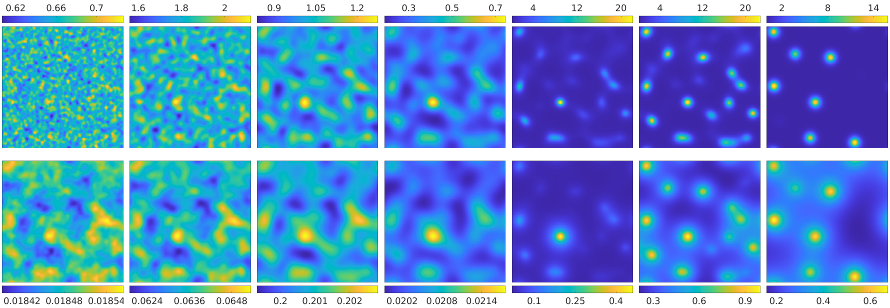
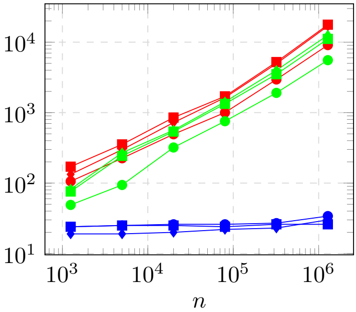
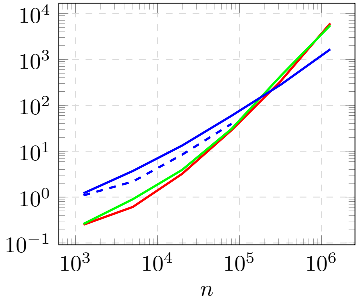


Figure: Example solution in 2D.



(a) Krylov iteration numbers



(b) Runtime in seconds

phipm — KIOPS — rk2expint iterative — rk2expint direct
exptAb_r. call 1 — exptAb_r. call 2 — exptAb_r. call 3

Figure: Scaling of 2D Gierer–Meinhardt example.

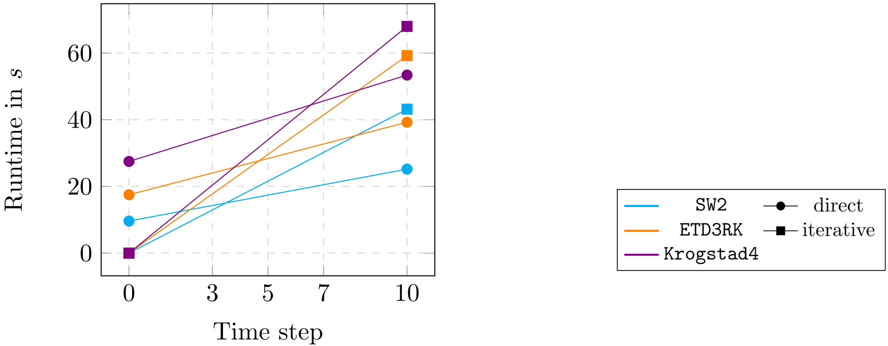
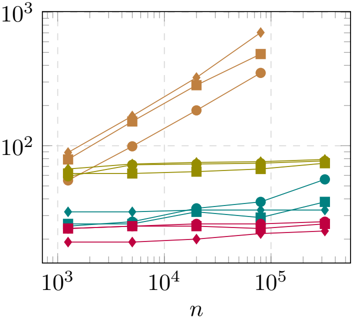
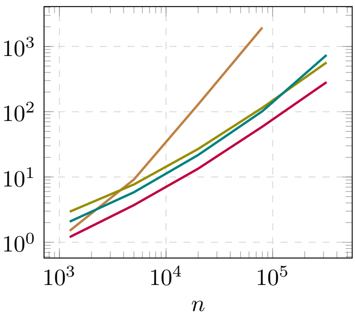


Figure: Comparison of direct and preconditioned iterative solvers.



(a) Krylov iteration numbers



(b) Runtime in seconds

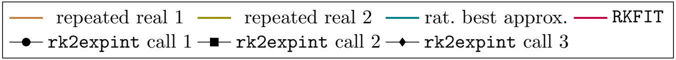


Figure: Comparison of different poles.

Numerical experiments

Allen–Cahn on networks

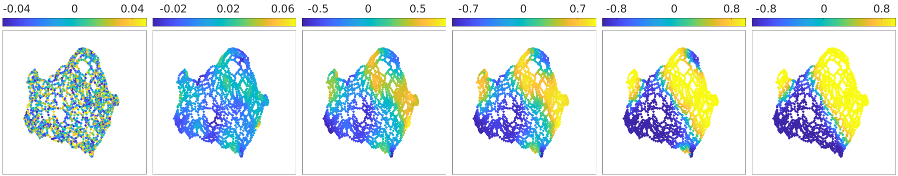


Figure: Example solution on *minnesota* network.

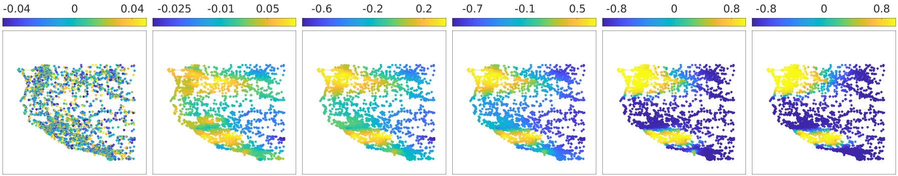
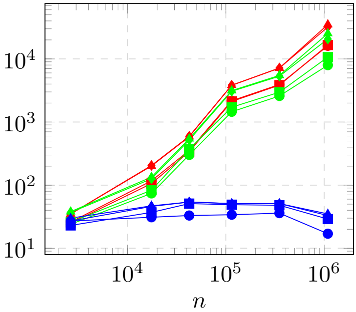
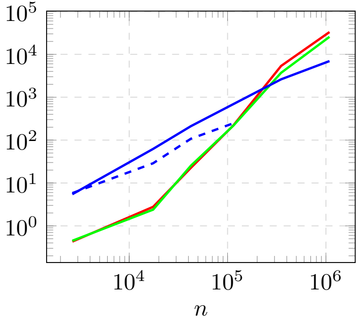


Figure: Example solution on *US roads (subset)* network.



(a) Krylov iteration numbers



(b) Runtime in seconds

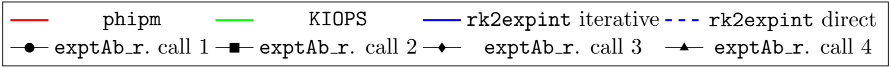
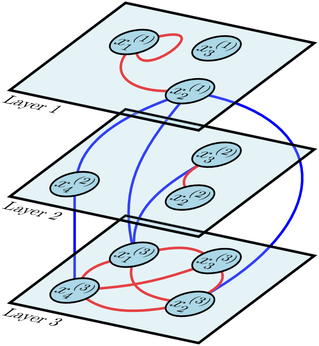


Figure: Scaling of Allen–Cahn on networks.

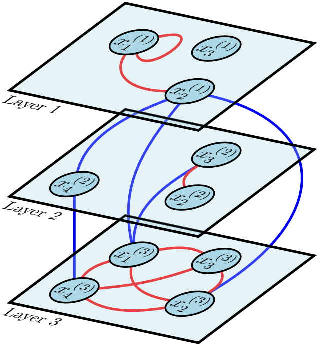
Exploiting algebraic structure

- A multilayer graph $\mathcal{G} = (\mathcal{V}^{(1)}, \ldots, \mathcal{V}^{(L)}, \mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(L)}, \widetilde{\mathcal{E}})$ with L layers consist of
  - L vertex sets $\mathcal{V}^{(l)}$,
  - L intra-layer edge sets $\mathcal{E}^{(l)}$,
  - one inter-layer edge set $\widetilde{\mathcal{E}}$.



- Layers can encode
  - different relationships,
  - different interactions,
  - different modes of transportation,
  - changes in time,
  - ...
- Special case: Multiplex networks. Inter-layer edges only between same nodes in different layers
- 2014: two survey papers [Kivelä et.al., 2014], [Boccaletti et.al., 2014] on multilayer graphs: Common framework for concepts from research areas including social, biological, physical, information and engineering sciences
- Since then: much research on generalizations of single-layer graph methods to the multilayer case

- A multilayer graph $\mathcal{G} = (\mathcal{V}^{(1)}, \ldots, \mathcal{V}^{(L)}, \mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(L)}, \widetilde{\mathcal{E}})$ with L layers consist of
  - L vertex sets $\mathcal{V}^{(l)}$,
  - L intra-layer edge sets $\mathcal{E}^{(l)}$,
  - one inter-layer edge set $\widetilde{\mathcal{E}}$.
- Layers can encode
  - different relationships,
  - different interactions,
  - different modes of transportation,
  - changes in time,
  - ...

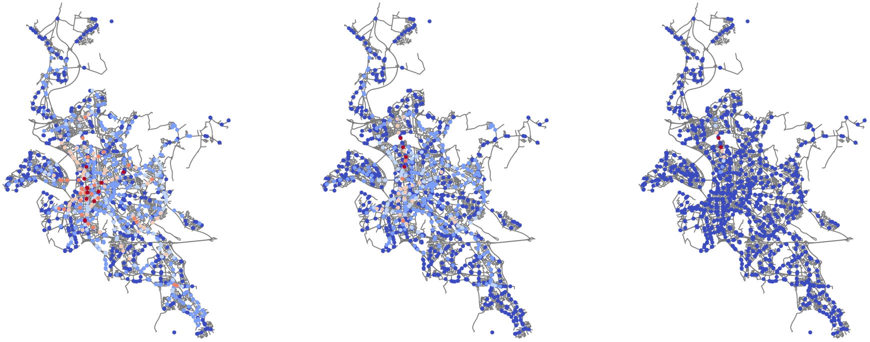


- Special case: Multiplex networks. Inter-layer edges only between same nodes in different layers
- 2014: two survey papers [Kivelä et.al., 2014], [Boccaletti et.al., 2014] on multilayer graphs: Common framework for concepts from research areas including social, biological, physical, information and engineering sciences
- Since then: much research on generalizations of single-layer graph methods to the multilayer case

Actors	broadcaster, $f(\boldsymbol{A})\boldsymbol{b}$				receiver, $f(\boldsymbol{A}^T)\boldsymbol{b}$			
	TC	KC	SC	SC_{res}	TC	KC	SC	SC_{res}
Vance Major	1	1	1	1	1	1	1	1
Adam Mullen	2	2	2	2	2	2	2	2
Kevin MacLeod	15	3	17	17	15	3	16	16
Gene Roddenberry	16	4	16	16	16	4	17	17
George Lucas	29	19	60	50	29	19	57	48
William Shatner	41	29	52	47	41	29	52	45
Jack Kirby	38	27	63	42	38	27	65	68
H.G. Wells	43	25	66	49	43	25	79	78
Leonard Nimoy	99	56	128	134	100	56	108	84
Jules Verne	113	67	160	183	114	67	117	60
Kate Mulgrew	106	92	102	116	106	92	104	119
James Cameron	118	71	147	161	117	71	149	160
Stephen King	150	91	235	248	149	91	235	248
Patrick Stewart	164	108	257	294	164	108	252	293

K. B., M. Stoll (2022). **Fast computation of matrix function-based centrality measures for layer-coupled multiplex networks**. *Physical Review E*, 105(3), 034305.
DOI:10.1103/PhysRevE.105.034305

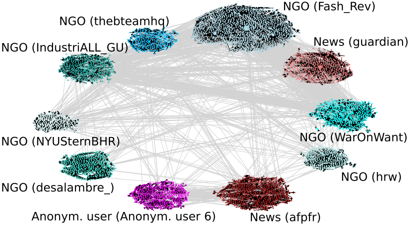
Exploiting algebraic structure



(a) $\alpha = 0.01/\lambda_{\max}$ (b) $\alpha = 0.75/\lambda_{\max}$ (c) $\alpha = 0.99/\lambda_{\max}$

K. B., M. Stoll (2021). **Orientations and matrix function-based centralities in multiplex network analysis of urban public transport**. *Applied Network Science*, 6, 90.
DOI:10.1007/s41109-021-00429-9

Exploiting algebraic structure



Rank	User	TC	User	KC
1	Fash_Rev	150	Fash_Rev	26.5
2	guardian	71	guardian	16.3
3	BritishVogue	62	BritishVogue	14.3
4	BoF	40	BoF	10.1
5	IndustriALL_GU	31	Presa_Diretta	7.9
6	thebteamhq	26	thebteamhq	7.8
7	WarOnWant	25	IndustriALL_GU	7.5
8	Presa_Diretta	25	WarOnWant	7.4
9	JasonMotlagh	24	KooyJan	7.4
10	KooyJan	22	AJEnglish	6.5

K. B., M. Wolter (2023). **A Twitter network and discourse analysis of the Rana Plaza collapse**. *Applied Network Science*, 8, 74. DOI:10.1007/s41109-023-00587-y

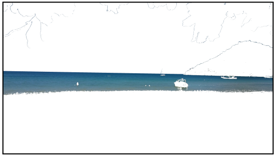
Exploiting algebraic structure

(a) Original image

(b) Tree

(c) Beach

(d) Prior labels

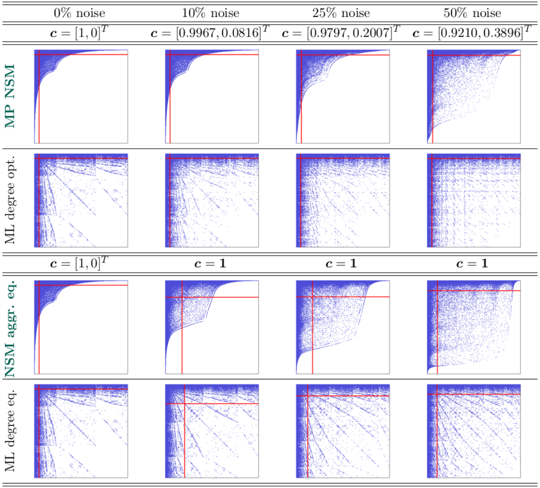


(e) Sea

(f) Sky

K. B., M. Stoll, T. Volkmer (2021). **Semi-supervised learning for aggregated multilayer graphs using diffuse interface methods and fast matrix vector products**. *SIAM Journal on Mathematics of Data Science*, 3(2), 758–785. DOI:10.1137/20M1352028

Exploiting algebraic structure



K. B., M. Stoll, F. Tudisco (2023). **A nonlinear spectral core-periphery detection method for multiplex networks**. *arXiv preprint*. DOI:10.48550/arXiv.2310.19697

- Setting: n nodes, L layers, $\boldsymbol{A}^{(l)}$ the adjacency matrix, $\boldsymbol{D}^{(l)} = \operatorname{diag}(\boldsymbol{A}^{(l)}\mathbf{1})$ the diagonal degree matrix, and $\boldsymbol{L}^{(l)} = \boldsymbol{D}^{(l)} - \boldsymbol{A}^{(l)}$ the graph Laplacian of layer l.
- Supra-Laplacian for multiplex networks:

$$\begin{aligned}\boldsymbol{L}_{\text{supra}} &= \boldsymbol{L}_{\text{intra}} + \omega \boldsymbol{L}_{\text{inter}} \\ &= \operatorname{blkdiag}\left[\boldsymbol{D}^{(1)}, \dots, \boldsymbol{D}^{(L)}\right] + \omega\left(\operatorname{diag}(\tilde{\boldsymbol{A}}\mathbf{1} \otimes \mathbf{1}) - \tilde{\boldsymbol{A}} \otimes \boldsymbol{I}\right),\end{aligned}$$

  with $\tilde{\boldsymbol{A}} \in \mathbb{R}^{L \times L}$ inter-layer coupling adjacency matrix and $\omega \in \mathbb{R}_{\geq 0}$ coupling parameter.
- ...

- Setting: n nodes, L layers, $\boldsymbol{A}^{(l)}$ the adjacency matrix, $\boldsymbol{D}^{(l)} = \text{diag}(\boldsymbol{A}^{(l)}\mathbf{1})$ the diagonal degree matrix, and $\boldsymbol{L}^{(l)} = \boldsymbol{D}^{(l)} - \boldsymbol{A}^{(l)}$ the graph Laplacian of layer l.
- Supra-Laplacian for multiplex networks:

$$\begin{aligned}\boldsymbol{L}_{\text{supra}} &= \boldsymbol{L}_{\text{intra}} + \omega \boldsymbol{L}_{\text{inter}} \\ &= \text{blkdiag}\left[\boldsymbol{D}^{(1)}, \ldots, \boldsymbol{D}^{(L)}\right] + \omega \left(\text{diag}(\tilde{\boldsymbol{A}}\mathbf{1} \otimes \mathbf{1}) - \tilde{\boldsymbol{A}} \otimes \boldsymbol{I}\right),\end{aligned}$$

  with $\tilde{\boldsymbol{A}} \in \mathbb{R}^{L \times L}$ inter-layer coupling adjacency matrix and $\omega \in \mathbb{R}_{\geq 0}$ coupling parameter.
- ...

Conclusion

- We applied adaptive rational Krylov subspace methods to the efficient evaluation of exponential Runge–Kutta integrators.
- This required
  - Optimal pole selection
  - Efficient solution of the sequences of shifted linear systems
  - An a-posteriori error estimate to rational Krylov approximations of $e^{h_i \widetilde{\boldsymbol{A}}} \widetilde{\boldsymbol{c}}$
- It enables
  - constant rat. Krylov iteration numbers w.r.t. the problem size (spectral radius of the discrete linear differential operator $\boldsymbol{A}$)
  - a near-linear scaling of the runtime
  - runtime gains for sufficiently large spectral radii of $\boldsymbol{A}$
- Left for later:
  - Multiplex network case
  - Inexact rational Krylov methods
  - Nonsymmetric problems, e.g., including advection
  - Other types of integrators (exp. Rosenbrock/EPIRK methods)

K. B., M. Stoll (2024). **Adaptive Rational Krylov Methods for Exponential Runge–Kutta Integrators**. SIAM Journal on Matrix Analysis and Applications, 45(1), p.744–770, 2024.

Thanks for your attention!

Conclusion

- We applied adaptive rational Krylov subspace methods to the efficient evaluation of exponential Runge–Kutta integrators.
- This required
  - Optimal pole selection
  - Efficient solution of the sequences of shifted linear systems
  - An a-posteriori error estimate to rational Krylov approximations of $e^{h_i \widetilde{\boldsymbol{A}}} \widetilde{\boldsymbol{c}}$
- It enables
  - constant rat. Krylov iteration numbers w.r.t. the problem size (spectral radius of the discrete linear differential operator $\boldsymbol{A}$)
  - a near-linear scaling of the runtime
  - runtime gains for sufficiently large spectral radii of $\boldsymbol{A}$
- Left for later:
  - Multiplex network case
  - Inexact rational Krylov methods
  - Nonsymmetric problems, e.g., including advection
  - Other types of integrators (exp. Rosenbrock/EPIRK methods)

K. B., M. Stoll (2024). **Adaptive Rational Krylov Methods for Exponential Runge–Kutta Integrators**. SIAM Journal on Matrix Analysis and Applications, 45(1), p.744–770, 2024.

Thanks for your attention!

Conclusion

- We applied adaptive rational Krylov subspace methods to the efficient evaluation of exponential Runge–Kutta integrators.
- This required
  - Optimal pole selection
  - Efficient solution of the sequences of shifted linear systems
  - An a-posteriori error estimate to rational Krylov approximations of $e^{h_i \widetilde{\boldsymbol{A}}} \widetilde{\boldsymbol{c}}$
- It enables
  - constant rat. Krylov iteration numbers w.r.t. the problem size (spectral radius of the discrete linear differential operator $\boldsymbol{A}$)
  - a near-linear scaling of the runtime
  - runtime gains for sufficiently large spectral radii of $\boldsymbol{A}$
- Left for later:
  - Multiplex network case
  - Inexact rational Krylov methods
  - Nonsymmetric problems, e.g., including advection
  - Other types of integrators (exp. Rosenbrock/EPIRK methods)

K. B., M. Stoll (2024). **Adaptive Rational Krylov Methods for Exponential Runge–Kutta Integrators**. SIAM Journal on Matrix Analysis and Applications, 45(1), p.744–770, 2024.

Thanks for your attention!

Conclusion

- We applied adaptive rational Krylov subspace methods to the efficient evaluation of exponential Runge–Kutta integrators.
- This required
  - Optimal pole selection
  - Efficient solution of the sequences of shifted linear systems
  - An a-posteriori error estimate to rational Krylov approximations of $e^{h_i \widetilde{\boldsymbol{A}}} \widetilde{\boldsymbol{c}}$
- It enables
  - constant rat. Krylov iteration numbers w.r.t. the problem size (spectral radius of the discrete linear differential operator $\boldsymbol{A}$)
  - a near-linear scaling of the runtime
  - runtime gains for sufficiently large spectral radii of $\boldsymbol{A}$
- Left for later:
  - Multiplex network case
  - Inexact rational Krylov methods
  - Nonsymmetric problems, e.g., including advection
  - Other types of integrators (exp. Rosenbrock/EPIRK methods)

K. B., M. Stoll (2024). **Adaptive Rational Krylov Methods for Exponential Runge–Kutta Integrators**. SIAM Journal on Matrix Analysis and Applications, 45(1), p.744–770, 2024.

Thanks for your attention!

- We applied adaptive rational Krylov subspace methods to the efficient evaluation of exponential Runge–Kutta integrators.
- This required
  - Optimal pole selection
  - Efficient solution of the sequences of shifted linear systems
  - An a-posteriori error estimate to rational Krylov approximations of $e^{h_i \widetilde{\boldsymbol{A}}} \widetilde{\boldsymbol{c}}$
- It enables
  - constant rat. Krylov iteration numbers w.r.t. the problem size (spectral radius of the discrete linear differential operator $\boldsymbol{A}$)
  - a near-linear scaling of the runtime
  - runtime gains for sufficiently large spectral radii of $\boldsymbol{A}$
- Left for later:
  - Multiplex network case
  - Inexact rational Krylov methods
  - Nonsymmetric problems, e.g., including advection
  - Other types of integrators (exp. Rosenbrock/EPIRK methods)

K. B., M. Stoll (2024). **Adaptive Rational Krylov Methods for Exponential Runge–Kutta Integrators**. SIAM Journal on Matrix Analysis and Applications, 45(1), p.744–770, 2024.

Thanks for your attention!

References I

- M. Hochbruck and A. Ostermann, Exponential integrators, Acta Numer., 19 (2010), pp. 209–286.
- R. Weiner, Linear-implizite Runge-Kutta-Methoden und ihre Anwendung, vol. 127, Springer-Verlag, 2013.
- S. M. Cox and P. C. Matthews, Exponential time differencing for stiff systems, J. Comput. Phys., 176 (2002), pp. 430–455.
- S. Krogstad, Generalized integrating factor methods for stiff PDEs, J. Comput. Phys., 203 (2005), pp. 72–88.
- Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal., 29 (1992), pp. 209–228.
- R. B. Sidje, Expokit: A software package for computing matrix exponentials, ACM Trans. Math. Software, 24 (1998), pp. 130–156.
- A. H. Al-Mohy and N. J. Higham, Computing the action of the matrix exponential, with an application to exponential integrators, SIAM J. Sci. Comput., 33 (2011), pp. 488–511.
- J. Niesen and W. M. Wright, Algorithm 919: A Krylov subspace algorithm for evaluating the φ-functions appearing in exponential integrators, ACM Trans. Math. Software, 38 (2012), pp. 1–19.
- S. Gaudreault, G. Rainwater, and M. Tokman, KIOPS: A fast adaptive Krylov subspace solver for exponential integrators, J. Comput. Phys., 372 (2018), pp. 236–255.
- M. Hochbruck and C. Lubich, On Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal., 34 (1997), pp. 1911–1925.
- S. Güttel, Rational Krylov approximation of matrix functions: Numerical methods and optimal pole selection, GAMM-Mitt., 36 (2013), pp. 8–31.
- A. Ruhe, Rational Krylov algorithms for nonsymmetric eigenvalue problems, in Recent Advances in Iterative Methods, Springer, 1994, pp. 149–164.
- M. Berljafa, S. Elsworth, and S. G üttel, A rational Krylov toolbox for MATLAB, Available at http://guettel.com/rktoolbox/, (2014).
- W. Cody, G. Meinardus, and R. Varga, Chebyshev rational approximations to e^{-x} in $[0, \infty)$ and applications to heat-conduction problems, J. Approx. Theory, 2 (1969), pp. 50–65.

References II

- A. Carpenter, A. Ruttan, and R. Varga, Extended numerical computations on the “1/9” conjecture in rational approximation theory, in Rational Approximation and Interpolation, Springer, 1984, pp. 383–411.
- E. Gallopoulos and Y. Saad, Efficient solution of parabolic equations by Krylov approximation methods, SIAM Journal on Scientific and Statistical Computing, 13 (1992), pp. 1236–1264.
- I. Moret and P. Novati, RD-rational approximations of the matrix exponential, BIT, 44 (2004), pp. 595–615.
- J. Van Den Eshof and M. Hochbruck, Preconditioning Lanczos approximations to the matrix exponential, SIAM J. Sci. Comput., 27 (2006), pp. 1438–1457.
- M. Berljafa and S. Güttel, Generalized rational Krylov decompositions with an application to rational approximation, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 894–916.
- M. Berljafa and S. Güttel, The RKFIT algorithm for nonlinear rational approximation, SIAM J. Sci. Comput., 39 (2017), pp. A2049–A2071.
- C. G. Petra, O. Schenk, and M. Anitescu, Real-time stochastic optimization of complex energy systems on high-performance computers, Computing in Science & Engineering, 16 (2014), pp. 32–42.
- C. G. Petra, O. Schenk, M. Lubin, and K. G ärtner, An augmented incomplete factorization approach for computing the Schur complement in stochastic optimization, SIAM J. Sci. Comput., 36 (2014), pp. C139–C162.
- Y. Notay, An aggregation-based algebraic multigrid method, Electron. Trans. Numer. Anal., 37 (2010), pp. 123–146.
- Y. Notay, Aggregation-based algebraic multigrid for convection-diffusion equations, SIAM J. Sci. Comput., 34 (2012), pp. A2288–A2316.
- A. Napov and Y. Notay, An algebraic multigrid method with guaranteed convergence rate, SIAM J. Sci. Comput., 34 (2012), pp. A1079–A1109.
- H. Nakao and A. S. Mikhailov, Turing patterns in network-organized activator–inhibitor systems, Nature Physics, 6 (2010), pp. 544–550.
- Kivelä, M. et.al. (2014) Multilayer networks. *Journal of Complex Networks*, 2(3), 203–271.
- Boccaletti, S. et.al. (2014) The structure and dynamics of multilayer networks. *Physics Reports*, 544(1), 1–122.