Numerische Mathematik (2023) 155:175–209
https://doi.org/10.1007/s00211-023-01372-w

Numerische
Mathematik

Improved ParaDiag via low-rank updates and interpolation

Daniel Kressner[1] · Stefano Massei[2] · Junli Zhu[3]

Received: 8 February 2023 / Revised: 14 June 2023 / Accepted: 7 August 2023 /
Published online: 20 September 2023

Abstract

This work is concerned with linear matrix equations that arise from the space-time discretization of time-dependent linear partial differential equations (PDEs). Such matrix equations have been considered, for example, in the context of parallel-in-time integration leading to a class of algorithms called ParaDiag. We develop and analyze two novel approaches for the numerical solution of such equations. Our first approach is based on the observation that the modification of these equations performed by ParaDiag in order to solve them in parallel has low rank. Building upon previous work on low-rank updates of matrix equations, this allows us to make use of tensorized Krylov subspace methods to account for the modification. Our second approach is based on interpolating the solution of the matrix equation from the solutions of several modifications. Both approaches avoid the use of iterative refinement needed by ParaDiag and related space-time approaches in order to attain good accuracy. In turn, our new algorithms have the potential to outperform, sometimes significantly, existing methods. This potential is demonstrated for several different types of PDEs.

The work of the S. Massei was supported by the INdAM-GNCS project "Metodi basati su matrici e tensori strutturati per problemi di algebra lineare di grandi dimensioni". The work of the J. Zhu has been supported by National Natural Science Foundation of China (Grant Nos. 11471150, 12161030) and China Scholarship Council (Contract No. 202006180060).

✉ Stefano Massei
stefano.massei@unipi.it

Daniel Kressner
daniel.kressner@epfl.ch

Junli Zhu
zhujl18@lzu.edu.c

[1] MATH-ANCHP, École Polytechnique Fédérale de Lausanne, Station 8, 1015 Lausanne, Switzerland

[2] Department of Mathematics, University of Pisa, Pisa, Italy

[3] School of Mathematics and Statistics, Lanzhou University, Lanzhou 730000, Gansu
, People's Republic of China

Mathematics Subject Classification 65F10 · 65F45 · 65M22 · 93C20

1 Introduction

This paper is concerned with the efficient numerical solution of matrix equations that arise from implicit time integration of large systems of ordinary differential equations (ODEs). More specifically, we treat generalized Sylvester equations of the form

$$AXB_2^T + MXB_1^T = F, \tag{1}$$

where $F \in \mathbb{R}^{n\times n_t}$ and the columns of the unknown matrix $X \in \mathbb{R}^{n\times n_t}$ contain the approximate solution of the ODE at n_t time steps with constant step size. The matrices $A, M \in \mathbb{R}^{n\times n}$ are large and sparse, as they arise, for example, from the finite difference/element spatial discretization of a time-dependent partial differential equation (PDE). The matrices $B_1, B_2 \in \mathbb{R}^{n_t\times n_t}$ are banded Toeplitz matrices and contain the coefficients of the time integrator. We assume throughout this work that M and B_2 are invertible.

Vectorizing (1) and using the Kronecker product $\otimes$, we obtain the equivalent $nn_t \times nn_t$ linear system

$$(B_2 \otimes A + B_1 \otimes M)\,\mathbf{x} = \mathbf{f}, \tag{2}$$

where $\mathbf{x} = \text{vec}(X)$ and $\mathbf{f} = \text{vec}(F)$. While the theoretical properties of generalized Sylvester equations are well understood and various numerical solution strategies have been developed [6, 46], the structure of the matrix coefficients in (1) comes with particular challenges that are addressed in this work.

Illustrative example. To illustrate (1) and motivate our developments, we consider the ODE

$$\begin{cases} M\dot{\mathbf{u}}(t) + A\mathbf{u}(t) = \mathbf{f}(t), & t \in (0, T], \\ \mathbf{u}(0) = \mathbf{x}_0, \end{cases} \tag{3}$$

with $\mathbf{u}(t) \in \mathbb{R}^n$. When such an ODE arises from the finite element discretization of a linear time-dependent PDE then A and M correspond to the stiffness matrix and the mass matrix, respectively, and M is symmetric positive definite. When using a finite difference discretization, we have $M = I_n$. The implicit Euler method with step size $\Delta t = T/n_t$ applied to (3) requires the successive solution of the linear systems

$$(\Delta t^{-1}M + A)\mathbf{x}_k = \mathbf{f}_k + \Delta t^{-1}M\mathbf{x}_{k-1}, \qquad k = 1, \ldots, n_t, \tag{4}$$

where $\mathbf{x}_k \approx \mathbf{u}(k \cdot \Delta t)$. Stacking these equations yields an $nn_t \times nn_t$ linear system of the form

$$\begin{bmatrix} \Delta t^{-1}M + A & & & \\ -\Delta t^{-1}M & \Delta t^{-1}M + A & & \\ & \ddots & \ddots & \\ & & -\Delta t^{-1}M & \Delta t^{-1}M + A \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{n_t} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 + \Delta t^{-1}M\mathbf{x}_0 \\ \mathbf{f}_2 \\ \vdots \\ \mathbf{f}_{n_t} \end{bmatrix}.$$

The system matrix can be rewritten in the form $B_2 \otimes A + B_1 \otimes M$ from (2) by setting

$$B_1 = \frac{1}{\Delta t} \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \qquad B_2 = I_{n_t}. \tag{5}$$

In turn, the linear system can be rephrased as a matrix equation of the form (1).

Taking a closer look at (2), we identify the factors that have a role in determining such a peculiar structure. The Kronecker structure of the coefficient matrix is due to the time-independence of the coefficients of the linear ODE (3). The matrices B_1, B_2 are Toeplitz because of the use of a constant step size. Indeed, this property is maintained when replacing implicit Euler by implicit multistep methods [27] such as BDF (backward differentiation formula). When using Runge–Kutta integrators [27], we still get banded Toeplitz matrices B_1, B_2 but – as we will see in Sect. 4 – we have to embed A, M into larger matrices in order to arrive at a matrix equation of the form (1).

Existing work. Classical time integration approaches for ODEs proceed by solving the discretized equations, like (3), sequentially first for $k = 1$, then for $k = 2$, and so on. This limits the potential for parallelizing the implementation of the integrator. The Parareal algorithm [22, 33] avoids this limitation by combining, iteratively, coarse (cheap) time steps that are performed sequentially with fine (expensive) time steps that are performed in parallel. There have been many developments and modifications of Parareal during the last two decades; we refer to [15, 21] for overviews and a historical perspective. Our work relates to the so called ParaDiag algorithms [20], which proceed somewhat differently from Parareal by explicitly exploiting the structure of the matrix Eq. (1). In passing, we mention that linear ODEs with constant coefficients, like the model problem (3), can be turned into an independent set of linear systems by combining an integral transform, like the Laplace transform, with quadrature for the numerical evaluation of the inverse transform; see [15, Sec. 5.5] and the references therein.

The basic idea of ParaDiag is simple: If $B_2^{-1}B_1$ can be diagonalized by a similarity transformation then (1) decouples into n_t linear systems, which can be solved in parallel. The pitfall of this approach is that the most straightforward choice of time steps, constant time steps, leads to non-diagonalizable matrices; indeed, the matrix $B_2^{-1}B_1$ in (5) is one big Jordan block. When choosing a geometrically increasing sequence of time steps, such as $\Delta t_j = \Delta t_1 \tau^{j-1}$ for some $\tau > 1$ then $B_2^{-1}B_1$ has mutually distinct eigenvalues and thus becomes diagonalizable [35]. However, even when using a second order method in time, e.g., the Crank-Nicolson method, the condition number of the transformation matrix grows rapidly as n_t increases, which severely limits the number of time steps [18, 19].

A second class of ParaDiag algorithms allows for uniform time steps and perturbs B_1, B_2 to enforce diagonalizability. More specifically, the perturbed system matrix takes the form

$$P_\alpha = C_2^{(\alpha)} \otimes A + C_1^{(\alpha)} \otimes M, \tag{6}$$

where $C_1^{(\alpha)}$, $C_2^{(\alpha)}$ are Strang-type [9, 38, 48] α-circulant matrices[1] constructed from the Toeplitz matrices B_1, B_2. This matrix can be used within a stationary iteration $P_\alpha \mathbf{x}^{(j)} = (P_\alpha - B_2 \otimes A - B_1 \otimes M)\mathbf{x}^{(j-1)} + \mathbf{f}$ [34] or as a preconditioner for GMRES [10, 37] applied to (2). In the solution of linear systems with P_α one takes advantage of the fact that the matrices $C_1^{(\alpha)}$, $C_2^{(\alpha)}$ are simultaneously diagonalized by a scaled Fourier transform:

$$\Omega^* D_\alpha C_1^{(\alpha)} D_\alpha^{-1} \Omega = \operatorname{diag}(\lambda_{1,1}, \dots, \lambda_{1,n_t}),$$
$$\Omega^* D_\alpha C_2^{(\alpha)} D_\alpha^{-1} \Omega = \operatorname{diag}(\lambda_{2,1}, \dots, \lambda_{2,n_t}),$$

where $\Omega \in \mathbb{C}^{n_t \times n_t}$ is the discrete Fourier transform matrix and $D_\alpha = \operatorname{diag}(1, \alpha^{\frac{1}{n_t}}, \dots, \alpha^{\frac{n_t-1}{n_t}})$. In turn, P_α is block diagonalized:

$$(\Omega^* D_\alpha \otimes I_n) P_\alpha (D_\alpha^{-1} \Omega \otimes I_n) = \begin{bmatrix} \lambda_{1,1} M + \lambda_{2,1} A & & \\ & \ddots & \\ & & \lambda_{1,n_t} M + \lambda_{2,n_t} A \end{bmatrix}.$$

Using the fast Fourier transform (FFT), this allows us to compute the solution of a linear system, with P_α by combining FFT with the (embarrassingly parallel) solution of n_t linear systems with the matrices $(\lambda_{1,j} M + \lambda_{2,j} A$, $j = 1, \dots, n_t$. This procedure, rephrased for the matrix equation corresponding to $P_\alpha \mathbf{x} = \mathbf{f}$, is summarized in Algorithm 1. Note that, even though the original ODE (3) features real matrices, Algorithm 1 requires the solution of *complex* linear systems, which is more expensive (often by a factor 2–4) than solving real linear systems. This disadvantage of having to pass to complex arithmetic is shared by all diagonalization based methods discussed in this work.

Algorithm 1 Solution of $AX(C_2^{(\alpha)})^T + MX(C_1^{(\alpha)})^T = F$

```
procedure FAST_DIAG_SOLVE($A, M, B_1, B_2, F, \alpha$)
    Compute eigenvalues $\lambda_{1,1}, \dots, \lambda_{1,n_t}, \lambda_{2,1}, \dots, \lambda_{2,n_t}$ by FFT.
    Compute $X \leftarrow F D_\alpha$ by column scaling and $X \leftarrow X\Omega^*$ by FFT.
    for $j = 1, \dots, n_t$ do
        Solve linear system $X(:,j) \leftarrow (\lambda_{1,j} M + \lambda_{2,j} A)^{-1} X(:,j)$
    end for
    Compute $X \leftarrow X\Omega$ by FFT and $X \leftarrow X D_\alpha^{-1}$ by column scaling.
end procedure
```

[1] On a continuous-time level this strategy can viewed as turning the initial value problem into a boundary value problem with periodic boundary conditions and, hence, it is sometimes called *waveform relaxation* [23, 51].

The use of the preconditioner P_α can dramatically accelerate the convergence of an iterative solver for (2). This has been observed for parabolic problems in [37] when using P_α with $\alpha = \pm 1$ as a preconditioner in GMRES. In [20], wave propagation problems (integrated in time with a leap-frog finite difference scheme) have been successfully treated by choosing a relatively small value for $0 < \alpha < 1$. Note, however, that α cannot be chosen arbitrarily small because otherwise the condition number of the matrix D_α (given by $\alpha^{(1-n_t)/n_t} \approx 1/\alpha$) explodes and, in turn, roundoff error impedes convergence. In practice, this means that the choice of α needs to strike a compromise and there is no choice of α that yields sudden convergence. In turn, several iterations are needed to attain good accuracy, which adds significant computational overhead and communication cost because every iteration calls Algorithm 1 once.

New contributions. In this paper, we propose two novel strategies that avoid the overhead incurred by iterative procedures for solving (2).

Our first method, presented in Sect. 2, exploits the trivial observation that $C_1^{(\alpha)} - B_1$ and $C_2^{(\alpha)} - B_2$ have very low rank. Thus, the Sylvester equation addressed by Algorithm 1 can be viewed as a low-rank modification of the original Eq. (1), which allows us to apply the low-rank update from [30] for linear matrix equations. This update corrects the output of Algorithm 1 by solving a Sylvester equation with the same coefficients as (1) but with a different right-hand side that has low rank. The main novelty of Sect. 2 is in the analysis of the low-rank structure of the solution to this equation and the development of efficient algorithms.

Our second method, presented in Sect. 3, takes an interpolation point of view: Whenever we solve the linear system $P_\alpha \mathbf{x} = \mathbf{f}$ we are actually evaluating a vector-valued function $\mathbf{x}(\alpha)$, depending on the complex parameter α, such that $\mathbf{x}(0)$ corresponds to the solution of (2). More precisely, this function can be efficiently and accurately evaluated on a scaled unit circle. We suggest to construct an approximation for $\mathbf{x}(0)$ via a linear combination of $\mathbf{x}(\alpha)$ evaluated at all dth order roots of a complex number for some small value of d. This strategy is highly parallelizable as it requires to solve d completely independent linear systems with coefficient matrices of the form P_α. Moreover, it applies also to the cases where the matrices $\delta B_j^{(\alpha)}$ are not low-rank, for example, when a time fractional differential operator is involved.

Section 4 extends our framework to implicit Runge–Kutta time integrators.

Section 5 is dedicated to a range of numerical experiments that highlight the advantages of our method for a broad range of situations.

2 Low-rank update approach

For a lower triangular banded Toeplitz matrix B, Strang's α-circulant preconditioner [38] is obtained from reflecting the lower band to the top right corner and multiplying it by α:

$$\begin{bmatrix} b_0 & & & & & \\ b_1 & b_0 & & & & \\ \vdots & \ddots & \ddots & & & \\ b_w & \ddots & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & b_w & \cdots & b_1 & b_0 \end{bmatrix} \longrightarrow \begin{bmatrix} b_0 & & & & & \\ b_1 & b_0 & & & & \\ \vdots & \ddots & \ddots & & & \\ b_w & \ddots & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & b_w & \cdots & b_1 & b_0 \end{bmatrix} + \alpha \begin{bmatrix} & & & b_w & \cdots & b_1 \\ & & & & \ddots & \vdots \\ & & & & & b_w \\ & & & & & \\ & & & & & \\ & & & & & \end{bmatrix}.$$

In particular, this implies that the rank of this modification is bounded by the width of the lower bandwidth. Applied to the time stepping matrices B_1, B_2, we obtain that

$$C_1^{(\alpha)} = B_1 + \delta B_1^{(\alpha)}, \qquad C_2^{(\alpha)} = B_2 + \delta B_2^{(\alpha)}$$

for certain low-rank matrices $\delta B_1^{(\alpha)}$, $\delta B_2^{(\alpha)}$. For instance, for the implicit Euler method (see (5)) we have $\delta B_1^{(\alpha)} = -\frac{\alpha}{\Delta t}\mathbf{e}_1\mathbf{e}_{n_t}^T$, where $\mathbf{e}_j$ denoting the jth unit vector of length n_t, and $\delta B_2^{(\alpha)} = 0$. Inspired by [30] we write the solution of (1) as $X = X_0 + \delta X$, where

$$AX_0(C_2^{(\alpha)})^T + MX_0(C_1^{(\alpha)})^T = F, \tag{7}$$

$$A\delta X B_2^T + M\delta X B_1^T = AX_0(\delta B_2^{(\alpha)})^T + MX_0(\delta B_1^{(\alpha)})^T. \tag{8}$$

After Eq. (7) is solved via Algorithm 1, the correction Eq. (8) becomes a generalized Sylvester equation in δX that has the same coefficients as (1) but a different right-hand side of rank at most $\text{rank}(\delta B_1^{(\alpha)}) + \text{rank}(\delta B_2^{(\alpha)})$. This remarkable property enables us to make use of low-rank solvers for linear matrix equations (see [46] for an overview) in order to approximate δX. Note that, the choice of $\alpha \neq 0$ has no influence on this strategy; for this reason we always choose $\alpha = 1$ when using the low-rank update approach. The resulting procedure is summarized in Algorithm 2.

Algorithm 2

```
procedure LOW_RANK_UPDATE($A, M, B_1, B_2, F$)
    $X_0 \leftarrow$ FAST_DIAG_SOLVE($A, M, B_1, B_2, F, 1$)
    Compute a low-rank factorization $UV^* = AX_0(\delta B_2^{(1)})^T + MX_0(\delta B_1^{(1)})^T$
    $\delta X \leftarrow$ LOW_RANK_SOLVER($A, M, B_1, B_2, U, V$)    ▷ Call to a low-rank solver for matrix equations
    return $X_0 + \delta X$
end procedure
```

Remark 1 It is not uncommon that the right-hand side matrix F of (1) has (numerically) low rank because, for example, the inhomogeneity $\mathbf{f}$ of the ODE (3) is constant or varies smoothly with respect to t. In such a situation, a low-rank solver can be applied directly to (1) and the following approach has been suggested by Palitta [39]: A block Krylov subspace method is used for reducing the spatial dimension combined with an application of the Shermann–Morrison–Woodbury formula to the resulting reduced equation. Our low-rank approach has the advantage that it is agnostic to properties of F, at the expense of having to solve (7). In principle, Palitta's method can be applied to solve the correction Eq. (8) but we found it more efficient for large n_t to use a two-sided approach that reduces the temporal dimension as well.

2.1 Solution of correction equation

In the following, we discuss the application of low-rank solvers to the correction Eq. (8). In view of the (assumed) invertibility of M and B_2 we replace (8) by the following Sylvester matrix equation:

$$\widetilde{A}\delta X + \delta X \widetilde{B}^T = C, \tag{9}$$

where

$$\widetilde{A} = M^{-1}A, \quad \widetilde{B} = B_2^{-1}B_1, \quad C = M^{-1}(AX_0(\delta B_2^{(1)})^T + MX_0(\delta B_1^{(1)})^T)B_2^{-T}$$

and $\mathrm{rank}(C) \leq k := \mathrm{rank}(\delta B_1^{(1)}) + \mathrm{rank}(\delta B_2^{(1)}) \ll \min\{n, n_t\}$. For given factorizations $\delta B_1^{(1)} = U_1 V_1^*$ and $\delta B_2^{(1)} = U_2 V_2^*$, a low-rank factorization $C = UV^*$, with $U \in \mathbb{R}^{n\times k}$ and $V \in \mathbb{R}^{n_t \times k}$, is obtained by setting

$$U = [M^{-1}AX_0V_2,\ X_0V_1], \qquad V = [B_2^{-1}U_2,\ B_2^{-1}U_1]. \tag{10}$$

Alternatively, when M is symmetric positive definite and its Cholesky factorization $M = LL^T$ is available, a possible symmetry of A is preserved by considering

$$\breve{A}\delta\breve{X} + \delta\breve{X}\breve{B}^T = \breve{C}, \tag{11}$$

with $\breve{A} = L^{-1}AL^{-T}$, $\breve{B} = \widetilde{B}$, $\breve{C} = L^{-1}(AX_0(\delta B_2^{(1)})^T + MX_0(\delta B_1^{(1)})^T)B_2^{-T}$, and $\delta\breve{X} = L^T\delta X$. In analogy to (10) an explicit rank-k factorization of $\breve{C}$ can be derived. In the rest of this section we focus on analyzing and solving (9) but we remark that our findings extend to (11) with minor modifications.

Various numerical methods have been developed to treat large-scale Sylvester equations with low-rank right-hand side, including the *Alternating Direction Implicit method* (ADI) [14, 40] and the *Rational Krylov Subspace Method* (RKSM) [5, 26, 42]. These methods compute approximate solutions of the form $\widetilde{\delta X} = WYZ^*$, where W and Z are bases of (block) rational Krylov subspaces generated with the matrices $\widetilde{A}$, $\widetilde{B}$ and starting block vectors U and V, respectively. More precisely, given two families $\xi = \{\xi_1, \dots, \xi_\ell\}$, $\psi = \{\psi_1, \dots, \psi_\ell\} \subset \mathbb{C}$ of so-called *shift parameters*, W and Z are chosen as bases of the following subspaces:

$$\begin{aligned} \mathcal{RK}(\widetilde{A}, U, \xi) &= \mathrm{span}\big((\xi_1 I - \widetilde{A})^{-1}U, \dots, (\xi_\ell I - \widetilde{A})^{-1}U\big), \\ \mathcal{RK}(\widetilde{B}, V, \psi) &= \mathrm{span}\big((\psi_1 I - \widetilde{B})^{-1}V, \dots, (\psi_\ell I - \widetilde{B})^{-1}V\big). \end{aligned} \tag{12}$$

When shifts are repeated, the matrix power is increased. For example, when a single shift ξ_1 is repeated ℓ times, one uses $\mathrm{span}\big((\xi_1 I - \widetilde{A})^{-1}U, \dots, (\xi_1 I - \widetilde{A})^{-\ell}U\big)$. We refer to the relevant literature [5, 13, 46] for a complete description of the computation of the factors W, Y, Z in these methods.

Let us comment on some implementation aspects of RKSM, which will be the method of choice in our implementation of Algorithm 2, line 4. In RKSM, the middle factor Y in the approximation $\widetilde{\delta X} = WYZ^*$ is obtained by solving a compressed matrix equation, which does not contribute significantly to the overall computational time as long as the dimensions of the Krylov subspaces remain modest. Computing W and Z requires to solve shifted linear systems with the matrices $\widetilde{A}$, $\widetilde{B}$; these operations can be performed without forming $M^{-1}A$ and $B_2^{-1}B_1$ explicitly, by means of the relations

$$(\xi_j I - \widetilde{A})^{-1}\mathbf{v} = (\xi_j M - A)^{-1}M\mathbf{v}, \qquad (\psi_j I - \widetilde{B})^{-1}\mathbf{v} = (\psi_j B_2 - B_1)^{-1}B_2\mathbf{v}.$$

In particular, we can still leverage properties like sparsity in A and M, or the bandedness of B_1, B_2, when solving shifted linear systems with $\widetilde{A}$, $\widetilde{B}$. Our implementation of RKSM relies on the Matlab toolbox `rk-toolbox` [7] for generating the factors W, Z. Finally and importantly, the selection of the shift parameters ξ, ψ strongly affects the convergence of RKSM and we discuss suitable choices in the following section.

2.2 Low-rank approximability of the correction equation

In this section, we analyze the numerical low-rank structure of the solution δX to the correction Eq. (9), which also yields suitable choices of shift parameters in RKSM discussed above.

By the Eckart-Young-Mirsky theorem, δX admits good low-rank approximations if its singular values $\sigma_j(\delta X)$ decay rapidly. Several works, including [1, 2, 41, 43], have established singular value decay bounds for solutions of Sylvester matrix equations. In particular, the framework presented in [4] applies to (9) and shows that [4, P. 323]

$$\sigma_{1+kj}(\delta X) \leq \min_{r(z)\in\mathcal{R}_{j,j}} \|r(\widetilde{A})\|_2 \|r(-\widetilde{B})^{-1}\|_2 \|\delta X\|_2, \tag{13}$$

where $\mathcal{R}_{j,j}$ denotes the set of rational functions having numerator and denominator of degree at most j. Choosing a good candidate for the rational function r is difficult, especially when $\widetilde{A}$ and/or $\widetilde{B}$ are non-normal. To circumvent this difficulty, we loosen the bound (13) by considering the numerical range $\mathcal{W}(\widetilde{A}) = \{x^*Ax \colon \|x\|_2 = 1\} \subset \mathbb{C}$. A result by Crouzeix and Palencia [11] states that

$$\|r(\widetilde{A})\|_2 \leq (1+\sqrt{2}) \max_{z\in\mathcal{W}(\widetilde{A})} |r(z)| \leq (1+\sqrt{2}) \max_{z\in E} |r(z)|,$$

where the constant $1+\sqrt{2}$ can be omitted when A is normal and the set $E \supseteq \mathcal{W}(\widetilde{A})$ is chosen to feature a simple geometry, for which the eventual rational approximation problem admits explicit solutions. Similarly, one has

$$\|r(-\widetilde{B})^{-1}\|_2 \le (1+\sqrt{2}) \max_{z\in F} |r(z)|^{-1} = (1+\sqrt{2})\big(\min_{z\in F} |r(z)|\big)^{-1}$$

for any set $F \subset \mathbb{C}$ with $F \supseteq \mathcal{W}(-\widetilde{B})$. Inserting these inequalities into (13), we arrive at

$$\frac{\sigma_{1+kj}(\delta X)}{\|\delta X\|_2} \le (1+\sqrt{2})^a Z_j(E,F), \qquad Z_j(E,F) := \min_{r(z)\in\mathcal{R}_{j,j}} \frac{\max_{z\in E}|r(z)|}{\min_{z\in F}|r(z)|},$$

where the exponent a satisfies $a = 0$ if both $\widetilde{A}$, $\widetilde{B}$ are normal, $a = 1$ if one of the two matrices is normal, and $a = 2$ otherwise.

The quantity $Z_j(E,F)$ and the related optimization problem are known in the literature as the *Zolotarev number* and *Zolotarev problem* for the sets E and F [4, 52]. Studying the Zolotarev problem is not only important from a theoretical but also from a computational point of view because identifying an extremal rational function $r_j^*(z)$ for $Z_j(E,F)$ allows for the fast solution of (9). Indeed, running j steps of ADI or RKSM with shift parameters given by the zeros and poles of $r_j^*(z)$ ensures a convergence rate not slower than the decay rate of $Z_j(E,F)$ [3, 32, 50]. A major complication of this construction is the need for choosing the sets E, F. On the one hand, it is desirable to choose E, F such that they enclose the numerical ranges of $\widetilde{A}$, $\widetilde{B}$ as tightly as possible. On the other hand, explicit solutions for $Z_j(E,F)$ are known only for a few selected geometries of E and F, including: two disjoint real intervals [52], the two connected components of the complement of an annulus [25], two disjoint discs [47]. Therefore, one usually encloses the numerical range by a disc or an interval (in the case of Hermitian matrices). In Sects. 2.2.1 and 2.2.2 below, we discuss two geometries in more detail: the case of two discs and the case of a disc and an interval. These enclosures will be used for generating the shift parameters in Algorithm 2 for most of our numerical examples.

2.2.1 Solution of the Zolotarev problem for two disjoint discs

Letting $\mathbb{B}(x,\rho)$ denote the closed disc with center $x \in \mathbb{C}$ and radius $\rho > 0$, we consider the Zolotarev problem for

$$E = \mathbb{B}(x_E,\rho_E), \qquad F = \mathbb{B}(x_F,\rho_F), \qquad |x_E - x_F| > \rho_E + \rho_F.$$

Starke [47] addressed the case $x_E, x_F \in \mathbb{R}$ by explicitly constructing a rational function $r_1^*(z) = \frac{z-q^*}{z-p^*}$ with $q^* \in E^{\mathrm{o}}$, $p^* \in F^{\mathrm{o}}$, and $|r_1^*(z)|$ being constant on ∂E and on ∂F. In view of the (generalized) *near-circularity criterion* [47, Theorem 3.1], this implies the optimality of the rational function $[r_1^*(z)]^j$ for $Z_j(E,F)$, $j \ge 1$.

The general case of two disjoint discs (with $x_E, x_F \in \mathbb{C}$ not necessarily on the real line) can be derived from Starke's result by relying on the invariance property of Zolotarev problems under Moebius transformations; see also [44, Example VIII.4.2]. For completeness and convenience of the reader, we provide the complete transformation that maps E, F to the complement of an annulus. More precisely, a Moebius map Φ is constructed such that $\Phi(F)$ is a closed disc centered at the origin and $\Phi(E)$

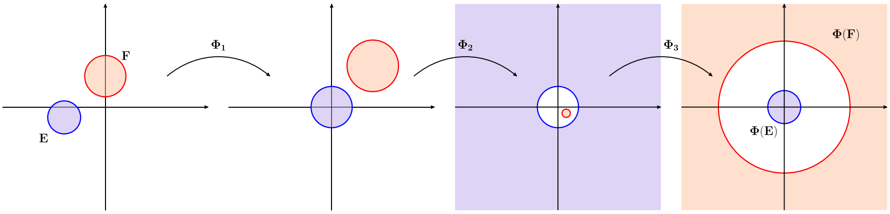

Fig. 1 Action of the Moebius map $\Phi = \Phi_3 \circ \Phi_2 \circ \Phi_1$ on the sets E (blue) and F (red)

is the complement of a larger open disc centered at the origin. We compose Φ from three simpler Moebius maps Φ_j, $j = 1, 2, 3$, as follows (see also Fig. 1):

- $\Phi_1(z) = \frac{z - x_E}{\rho_E}$ maps E to the unit disc while F remains a disjoint disc,
- $\Phi_2(z) = z^{-1}$ maps $\Phi_1(E)$ to the exterior of the unit disc and $\Phi_1(F)$ to a disc enclosed by the unit circle,
- $\Phi_3(z) = \frac{z-\alpha}{z-\beta}$ maps $\Phi_2(\Phi_1(F))$ to the exterior of a disc with center 0 and $\Phi_2(\Phi_1(E))$ to a disc centered at 0 and enclosed by $\partial\Phi_3(\Phi_2(\Phi_1(F)))$.

The parameters $\alpha, \beta \in \mathbb{C}$ are chosen as the so-called *common inverse points* for the circles $\partial\Phi_2(\Phi_1(E)) = \partial\mathbb{B}(0, 1)$ and $\partial\Phi_2(\Phi_1(F))$ [28, Section 4.2]. Setting $\Phi_2(\Phi_1(F)) =: \mathbb{B}(\widetilde{x}_F, \widetilde{\rho}_F)$ the values of α, β from satisfying the two equations

$$\alpha\overline{\beta} = 1, \qquad (\alpha - \widetilde{x}_F)\overline{(\beta - \widetilde{x}_F)} = \widetilde{\rho}_F^2.$$

Clearly, one solution is given by

$$\beta = \alpha^{-1}, \qquad \alpha = \left[|\widetilde{x}_F|^2 + 1 - \widetilde{\rho}_F^2 + \sqrt{(|\widetilde{x}_F|^2 + 1 - \widetilde{\rho}_F^2)^2 - 4|\widetilde{x}_F|^2} \right] / (2\overline{\widetilde{x}_F}). \tag{14}$$

Observe that, $\Phi_1(F)$ is a disc with center $\Phi_1(x_F) \neq 0$ and radius ρ_F/ρ_E. In particular, $\Phi_1(F)$ attains its maximum and minimum modulus at the elements

$$x_1 = \Phi_1(x_F) + \frac{\Phi_1(x_F)}{|\Phi_1(x_F)|}\frac{\rho_F}{\rho_E}, \qquad x_2 = \Phi_1(x_F) - \frac{\Phi_1(x_F)}{|\Phi_1(x_F)|}\frac{\rho_F}{\rho_E}.$$

Now, $\Phi_2(z) = z^{-1}$ maps x_1 and x_2 into the minimum and maximum modulus elements of $\mathbb{B}(\widetilde{x}_F, \widetilde{\rho}_F)$, respectively. This implies that $\widetilde{x}_F, \widetilde{\rho}_F$ take the following values:

$$\widetilde{x}_F = \frac{\Phi_2(x_1) + \Phi_2(x_2)}{2}, \qquad \widetilde{\rho}_F = \frac{|\Phi_2(x_1) - \Phi_2(x_2)|}{2}.$$

Together with (14), this allows us to compute α, β explicitly and evaluate Φ. The complete expressions for Φ and Φ^{-1} read as follows:

$$\Phi(z) = \frac{\rho_E - \alpha(z - x_E)}{\rho_E - \overline{\alpha}^{-1}(z - x_E)}, \qquad \Phi^{-1}(z) = \frac{(\overline{\alpha}^{-1}x_E + \rho_E)z - \alpha x_E - \rho_E}{\overline{\alpha}^{-1}z - \alpha}.$$

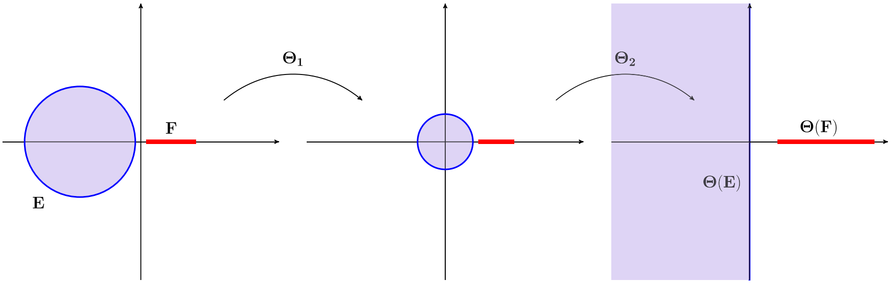

Fig. 2 Action of the Moebius map $\Theta = \Theta_2 \circ \Theta_1$ on the sets E (blue) and F (red)

By the near-circularity criterion [47, Theorem 3.1], an extremal rational function for the complement of an annulus centered at 0 is z^j, having the only pole ∞ and the only zero at 0. Thus, an optimal rational function for the original problem is given by $r_j^*(z) = \left(\frac{z-\Phi^{-1}(0)}{z-\Phi^{-1}(\infty)}\right)^j$. The pole p^* and zero q^* are given by

$$p^* = \Phi^{-1}(\infty) = x_E + \frac{\rho_E}{\overline{\alpha}^{-1}}, \qquad q^* = \Phi^{-1}(0) = x_E + \frac{\rho_E}{\alpha}. \tag{15}$$

Since Φ maps the boundaries of E and F to the boundaries of the annulus, it follows that the Zolotarev numbers are given by

$$\begin{aligned} Z_j(E, F) &= \eta^j, \\ \eta &= \frac{|\Phi(x_E + \rho_E)|}{|\Phi(x_F + \rho_F)|} = \left| \frac{(1-\alpha)(\rho_E - \overline{\alpha}^{-1}(x_F + \rho_F - x_E))}{(1-\overline{\alpha}^{-1})(\rho_E - \alpha(x_F + \rho_F - x_E))} \right| < 1. \end{aligned} \tag{16}$$

2.2.2 Quasi-optimal solution of the Zolotarev problem for a disc and an interval

Let us now consider the situation of a disc and a disjoint interval:

$$E = \mathbb{B}(x_E, \rho_E), \qquad F = [a, b], \qquad x_E \in \mathbb{R}, \quad x_E + \rho_E < a \text{ or } x_E - \rho_E > b.$$

Similarly to the previous section, we build a Moebius transformation that recasts $Z_j(E, F)$ as a Zolotarev problem for which a (quasi-optimal) solution is known. More specifically, we let $\Theta = \Theta_2 \circ \Theta_1$ where (see also Fig. 2):

- $\Theta_1(z) = \Phi_1(z)$ maps E to the unit disc while F remains a real disjoint interval,
- $\Theta_2(z) = \frac{z+1}{z-1}$ maps $\Theta_1(E)$ to the closed left half of the complex plane and $\Theta_1(F)$ to an interval $[\tilde{a}, \tilde{b}]$ of positive real numbers.

A straightforward calculation yields explicit expressions for Θ and its inverse:

$$\Theta(z) = \frac{z - x_E + \rho_E}{z - x_E - \rho_E}, \qquad \Theta^{-1}(z) = \frac{(x_E + \rho_E)z - x_E + \rho_E}{z - 1}.$$

By the maximum modulus principle, it suffices to consider the boundary of $\Theta(E)$ and thus the transformed Zolotarev problem takes the form

$$Z_j(\partial\Theta(E), \Theta(F)) = Z_j(\mathbf{i}\mathbb{R}, [\tilde{a}, \tilde{b}]) = \min_{r(z)\in\mathcal{R}_{j,j}} \frac{\max_{z\in\mathbf{i}\mathbb{R}} |r(z)|}{\min_{z\in[\tilde{a},\tilde{b}]} |r(z)|}. \tag{17}$$

Let $\tilde{r}_j$ denote the extremal rational function for the Zolotarev problem $Z_j([-\tilde{b}, -\tilde{a}], [\tilde{a}, \tilde{b}])$, that is, the imaginary axis is replaced by $[-\tilde{b}, -\tilde{a}]$. This function has been extensively studied in the literature; see [4, Sec. 3.1] and the references therein. In particular, it holds that

$$\tilde{r}_j(z) = \prod_{i=1}^{j} \frac{z + \tilde{\psi}_{j,i}}{z - \tilde{\psi}_{j,i}},$$

where the poles $\tilde{\psi}_{j,i} \in [\tilde{a}, \tilde{b}]$, $i = 1, \dots, j$, can be expressed in closed form via elliptic integrals. Note that $\tilde{r}_j(-z) = 1/\tilde{r}_j(z)$ and the maximum absolute value of $\tilde{r}_j$ is 1, which is assumed throughout the imaginary axis. By [4, Corollary 3.2], it holds that $Z_j([-\tilde{b}, -\tilde{a}], [\tilde{a}, \tilde{b}]) \le 4\eta^{-2j}$ with $\eta = \exp\left(\frac{\pi^2}{2\log(4\tilde{b}/\tilde{a})}\right)$. Inserting $\tilde{r}_j$ into (17) gives the following upper bound for $Z_j(\mathbf{i}\mathbb{R}, [\tilde{a}, \tilde{b}])$:

$$\delta_j := \frac{\max_{z\in\mathbf{i}\mathbb{R}} |\tilde{r}_j(z)|}{\min_{z\in[\tilde{a},\tilde{b}]} |\tilde{r}_j(z)|} = \frac{1}{\min_{z\in[\tilde{a},\tilde{b}]} |\tilde{r}_j(z)|} = \sqrt{Z_j([-\tilde{b}, -\tilde{a}], [\tilde{a}, \tilde{b}])} \le 2\eta^{-j}. \tag{18}$$

It has been shown in [12, 31] that $\tilde{r}_j$ satisfies the necessary optimality conditions for (17) and that it is within a factor 2 of the optimal solution: $\delta_j \le 2Z_j(\mathbf{i}\mathbb{R}, [\tilde{a}, \tilde{b}])$. To the best of our knowledge, it is still an open question whether $\tilde{r}_j$ is, in fact, an extremal rational function for (17). In summary, we have

$$Z_j(E, F) \le 2\exp\left(\frac{-j\pi^2}{2\log(4\tilde{b}/\tilde{a})}\right) \tag{19}$$

with a quasi-optimal solution for $Z_j(E, F)$ given by

$$\widehat{r}_j(z) = \prod_{i=1}^{j} \frac{z - \Theta^{-1}(-\tilde{\psi}_{j,i})}{z - \Theta^{-1}(\tilde{\psi}_{j,i})}. \tag{20}$$

The poles of the rational function (19) are not nested, that is, the poles of $\widehat{r}_j(z)$ are different from the ones of $\widehat{r}_{j+1}(z)$. This is a disadvantage when using the poles of $\widehat{r}_j(z)$ in an iterative method (such as ADI or RKSM) where usually the number of steps (the parameter j) is not known a priori. Choosing j adaptively would require to restart the method from scratch whenever j is changed. In such a situation it is preferable to use a sequence of rational functions with nested poles. The method of *equidistributed* *sequences* (EDS) [12, Section 4] allows the generation of a nested sequence of poles ξ_i, $i = 1, 2, \ldots$, such that the rational functions $\prod_{i=1}^{j} \frac{z+\xi_i}{z-\xi_i}$ have asymptotically optimal convergence rates for $Z_j([-\tilde{b}, -\tilde{a}], [\tilde{a}, \tilde{b}])$ as j increases, see also [36, Section 3.5]. In analogy with (20), the nested asymptotically optimal zeros and poles for $Z_j(E, F)$ are given by $\Theta^{-1}(-\xi_i)$ and $\Theta^{-1}(\xi_i)$, respectively. In the following example and more generally in Sect. 5, we demonstrate that the shift parameters computed by means of EDS yield convergence rates very close to the ones obtained with the optimal solution of $Z_j([-\tilde{b}, -\tilde{a}], [\tilde{a}, \tilde{b}])$.

Example 1 Let us consider the following $1D$ heat equation from [16, Section 7.1]:

$$\begin{cases} u_t = u_{xx} + f(x,t), & x \in \Omega := [0,1], t \in [0,1] \\ u \equiv 0, & \text{on } \partial\Omega, \\ u(x,0) = 4x(1-x), & \text{at } t = 0, \\ f(x,t) = h\max\{1 - |c(t) - x|/w, 0\}, & c(t) = \frac{1}{2} + (\frac{1}{2} - w)\sin(2\pi t), \end{cases} \tag{21}$$

with $w = 0.05$ and $h = 100$. Discretizing in space with second order central differences and in time with the implicit Euler method on a uniform $n \times n_t$ grid yields a matrix equation of the form $AX + XB_1^T = F$ with $A \in \mathbb{R}^{n\times n}$ and $B_1 \in \mathbb{R}^{n_t \times n_t}$ are the matrices of centered and backward differences with respect to the x and t variables. For this example we set $n = n_t = 1000$ and rescale the equation by multiplying both sides with Δt. We focus on solving the correction equation

$$A\delta X + \delta X B_1^T = -X_0 \mathbf{e}_n \mathbf{e}_1^T, \tag{22}$$

where X_0 satisfies $AX_0 + X_0(C_1^{(1)})^T = F$. For this purpose, we use RKSM based on one of the rational functions constructed above. The poles and zeros of the rational function determine the shifts in the rational Krylov subspaces for A and B_1, respectively, see (12). We consider the following choices:

- `2DISCS` single shifts p^* and q^* resulting from the Zolotarev problems with 2 discs, see (15);
- `ZOL-DI` optimal shifts resulting from the Zolotarev problem with a disc and an interval, see (20);
- `EDS` nested, asymptotically optimal shifts resulting from the Zolotarev problem with a disc and an interval; the computation of equidistributed shifts is described in [36, Section 3.5].
- `EK` alternating shifts 0 and ∞, corresponding to the *extended Krylov method* [45].

The numerical ranges of the matrices A, B_1, which determine the sets E, F, are known analytically: $\mathcal{W}(B_1) = \{z : |z+1| \leq \cos(\pi/(n_t+1))\}$ and $\mathcal{W}(A) = [\lambda_{\min}, \lambda_{\max}]$ with $\lambda_{\min} = (2 - 2\cos(\pi/(n+1)))\frac{(n+1)^2}{n_t}$ and $\lambda_{\max} = (2 - 2\cos(n\pi/(n+1)))\frac{(n+1)^2}{n_t}$. We can directly use these sets for generating the shift parameters in `EDS` and `ZOL-DI`, while `2DISCS` uses the (suboptimal) inclusion $\mathcal{W}(A) \subset \{z : |z - (\lambda_{\max} + \lambda_{\min})/2| \leq (\lambda_{\max} - \lambda_{\min})/2\}$.

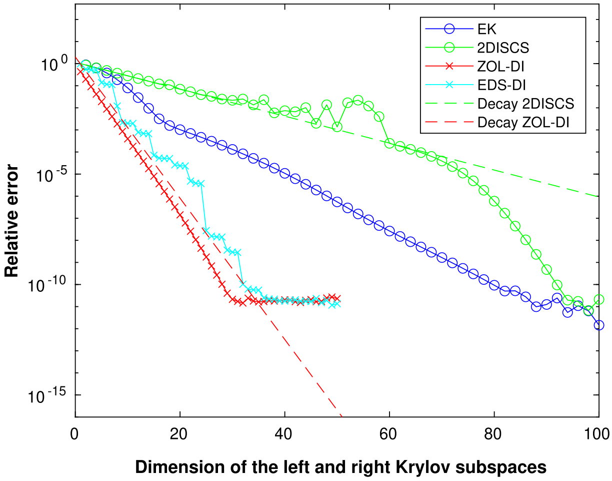

Fig. 3 Relative error convergence histories of RKSM for (9), in the case of the 1D heat equation (Eq. (22)), for various choices of the shift parameters

Letting $\widetilde{\delta X}$ denote the approximate solution produced by one of the choices above, Fig. 3 reports the relative error history $\|\widetilde{\delta X} - \delta X\|_2/\|\delta X\|_2$, where δX is a benchmark solution computed via the `lyap` command of Matlab, as the dimension of the rational Krylov subspaces increases. Finally, we report the relative error associated with the truncated singular values decomposition of δX to show the best error attainable. The results are shown in Fig. 3 together with the decay bounds (16) (`Decay 2DISCS`) and (19) (`Decay ZOL-DI`). The observed convergence of the residuals clearly highlights the advantage of choosing shifts via the Zolotarev problem with a disc and an interval, either optimally (`ZOL-DI`) or asymptotically optimally (`EDS`).

2.3 Cost analysis of Algorithm 2

In order to compare with our second approach in the next section, we briefly discuss the cost of Algorithm 2 as well as the potential for carrying out (embarrassingly) parallel computations. Let $\mathcal{C}_{\text{sys}}$ denote the maximum cost of solving a lineaear system involving a linear combinationr system with a linear combination of the matrices A and M, for a constant number of right-hand sides, respectively; note that, $\mathcal{C}_{\text{sys}}$ is also an upper bound for the cost of solving with A or M, individually. The cost of a matrix–vector operation with the banded matrices B_1, B_2 is $\mathcal{O}(n_t)$, which is negligible. We may assume $k = \mathcal{O}(1)$, which holds for all time discretization schemes discussed in this work. We assume that RKSM converges after ℓ iterations to fixed accuracy and that the shift parameters are all different, as it happens, for instance, in `EDS`. If A is symmetric positive definite and its condition number grows polynomially with n (as, e.g., for finite difference or finite element discretizations of elliptic operators) then the bound (19) predicts $\ell = \mathcal{O}(\log n)$. RKSM needs to solve $\mathcal{O}(\ell)$ linear systems at cost $\mathcal{C}_{\text{sys}}$ each for generating the left factor W of the approximate solution $\delta X \approx WYZ^*$. The (re)orthogonalization of the bases W, Z accounts for another $\mathcal{O}((n + n_t)\ell^2)$

operations. As we expect ℓ to be small, the cost for computing Y as the solution of the compressed equation is negligible. When $M \neq I_n$, computing the low-rank factorization at line 3 requires an additional constant number of linear systems solves. By adding the cost of Algorithm 1 called in line 2 of Algorithm 2 we obtain the overall complexity

$$\mathcal{O}\left((n_t + \ell)\mathcal{C}_{\text{sys}} + (n + n_t)\ell^2 + nn_t \log(n_t)\right).$$

In terms of parallelization, note that the linear systems solves of Algorithm 1 are embarrassingly parallel on up to n_t cores. In this situation, the cost reduces to $\mathcal{O}(\ell\mathcal{C}_{\text{sys}} + (n + n_t)\ell^2 + nn_t \log(n_t))$ for each of the n_t cores. A further reduction can be obtained by making use of parallel implementations of RKSM [8] and FFT.

3 An evaluation interpolation approach

As pointed out in the introduction, existing ParaDiag algorithms replace (1) with

$$AX(C_2^{(\alpha)})^T + MX(C_1^{(\alpha)})^T = F \tag{23}$$

for fixed α, where $C_1^{(\alpha)}, C_2^{(\alpha)}$ are α-cyclic matrices. Solving Eq. (23) can be carried out efficiently with Algorithm 1 and serves as a preconditioner in a Krylov method or a stationary iteration.

In this section we propose a new method that approximates the solution of (1) by combining solutions of the matrix Eq. (23) for different values of α. Given $\rho > 0$, we let $\mathcal{X}_\rho$ denote the matrix-valued function that maps $z \in \mathbb{C}$ to the solution $X = \mathcal{X}_\rho(z)$ of (23) for $\alpha = \rho z$. In particular $\mathcal{X}_\rho(0)$ is the desired solution of (1). In the following, we assume that $\mathcal{X}_\rho(z)$ is analytic in the disc $\{z : |z| < \widehat{\rho}\}$ for some $\widehat{\rho} > 1$; see Example 2 below for an illustration of this assumption.

Our approach proceeds by interpolating $\mathcal{X}_\rho(z)$ at the dth roots of unity for some integer d, $\omega_j = \exp(2\pi \mathbf{i} j/d)$ for $j = 0, \ldots, d-1$, with the interpolation data $\mathcal{X}_\rho(\omega_j)$ computed by Algorithm 1. It is well known (see, e.g., [49]) that the coefficients of the interpolating polynomial $\widetilde{\mathcal{X}}^{(d)}(z) = \sum_{j=0}^{d-1} \widetilde{X}_j^{(d)} z^j$ can be obtained from the interpolation data by applying the inverse Fourier transform. In particular, we obtain

$$\mathcal{X}_\rho(0) \approx \widetilde{\mathcal{X}}^{(d)}(0) = \widetilde{X}_0^{(d)} = \big(\mathcal{X}_\rho(\omega_0) + \cdots + \mathcal{X}_\rho(\omega_{d-1})\big)/d.$$

The described procedure is summarized in Algorithm 3. The analysis of the approximation error for such a trigonometric interpolation has a well established theory. The following result, retrieved by applying Theorem 12.1 from [49] to each entry, shows that $\widetilde{X}_0^{(d)}$ converges exponentially fast to the solution of (1) as d increases with the convergence rate depending on the radius of analyticity $\widehat{\rho}$.

Theorem 1 *Let $\mathcal{X}_\rho(z)$ be analytic on $\{z : |z| < \widehat{\rho}\}$ with $\widehat{\rho} > 1$ and choose $R \in (1, \widehat{\rho})$. Then the following holds for the entries of the approximation $\widetilde{X}_0^{(d)}$ returned by*

Algorithm 3:

$$|(\mathcal{X}_\rho(0))_{ij} - (\widetilde{X}_0^{(d)})_{ij}| \le \frac{\max_{|z|=R} |(\mathcal{X}_\rho(z))_{ij}|}{R^d - 1}, \qquad i = 1, \ldots n, \quad j = 1, \ldots, n_t.$$

Algorithm 3

```
procedure EV_INT($A, M, B_1, B_2, F, \rho, d$)
    for $j = 0, \ldots, d-1$ do
        $\omega_j \leftarrow \exp(2\mathbf{i}j\pi/d)$
        $X_j \leftarrow$ FAST_DIAG_SOLVE($A, M, B_1, B_2, F, \rho\omega_j$)
    end for
    return $\widetilde{X} \leftarrow (X_0 + \cdots + X_{d-1})/d$
end procedure
```

We remark that it is possible to implement an adaptive doubling strategy for the choice of the parameter d. Whenever the accuracy of $\widetilde{X}_0^{(d)}$ is unsatisfactory, we can consider the computation of $\widetilde{X}_0^{(2d)}$. Since a dth root of the unity is also a $2d$th root of the unity, this has the advantage that half of the evaluations of $\mathcal{X}_\rho(z)$ that we need in Algorithm 3 have already been computed at the previous iteration. As stopping criterion, one can verify that the norm of the difference between two consecutive approximations is smaller than a certain threshold. However, in all numerical experiments considered in this work we observe that the choices $d = 2, 3$ already provide sufficient accuracy and there is little need for an adaptive choice for d.

Example 2 Let us consider the matrix equation $AX + XB_1^T = F$ arising from the 1D heat equation of Example 1 with $n = n_t$. Then $C_1^{(\rho z)} = B_1 - \rho z \mathbf{e}_1 \mathbf{e}_n^T$ with the matrix B_1 from (5) multiplied with Δt. The eigendecomposition of $C_1^{(\rho z)}$ is given by

$$C_1^{(\rho z)} = \begin{bmatrix} 1 & & & \\ & (\rho z)^{-\frac{1}{n}} & & \\ & & \ddots & \\ & & & (\rho z)^{\frac{1-n}{n}} \end{bmatrix} \Omega \begin{bmatrix} 1-(\rho z)^{\frac{1}{n}} & & & \\ & 1-(\rho z)^{\frac{1}{n}}\zeta & & \\ & & \ddots & \\ & & & 1-(\rho z)^{\frac{1}{n}}\zeta^{n-1} \end{bmatrix}$$
$$\times \Omega^* \begin{bmatrix} 1 & & & \\ & (\rho z)^{\frac{1}{n}} & & \\ & & \ddots & \\ & & & (\rho z)^{\frac{n-1}{n}} \end{bmatrix},$$

where $\zeta = \exp(2\mathbf{i}\pi/n))$ and Ω is the discrete Fourier transform. In particular, the eigenvalues of $C_1^{(\rho z)}$ are contained in a disc with center 1 and radius $|\rho z|^{1/n}$. Since A is symmetric positive definite, the spectra of A and $-C_1^{(\rho z)}$ stay separated as long as $|z| < \rho^{-\frac{1}{n}} + \lambda_{\min}(A)$. This implies that $X_\rho(z)$ is well-defined and, as a rational function, also analytic on the open disc with radius $\rho^{-\frac{1}{n}} + \lambda_{\min}(A)$ and hence the assumptions of Theorem 1 can be satisfied as long as $\rho \le 1$. Assuming $|\rho z| \le 1$, the 2-norm condition number of the eigenvector matrix for $C_1^{(\rho z)}$ is bounded by $1/|\rho z|$ and, in turn, $\|\mathcal{X}_\rho(z)\|_F \le |\rho z \lambda_{\min}|^{-1} \|F\|_F$. Choosing $R = 1/\rho$ in Theorem 1, we thus obtain

$$\|X_0 - \widetilde{X}_0^{(d)}\|_F \leq \frac{1}{\lambda_{\min}(\rho^{-d} - 1)}\|F\|_F.$$

Hence, we expect to obtain very fast convergence with decreasing ρ and/or increasing d. Note that ρ cannot be chosen too small because otherwise roundoff error, due to the ill-conditioned eigenvector matrix of $X(\rho\omega_j)$, starts interfering with the interpolation error. Both statements are confirmed by numerical experiments. Table 1 shows relative residual Res $:= \|A\widetilde{X}_0 + \widetilde{X}_0 B_1^T - F\|_F / \|F\|_F$ for the matrix $\widetilde{X}_0$ computed by Algorithm 3 obtained with different values of n, ρ, d. The results indicate that excellent accuracy can already achieved for $d = 2$ when choosing ρ properly.

3.1 Cost analysis of Algorithm 3

In this section we maintain the assumptions and notation of Sect. 2.3. In particular, $\mathcal{C}_{\text{sys}}$ denotes the cost of solving linear systems with a linear combination of the matrices A and M. The asymptotic complexity of Algorithm 3 is dominated by the d calls to Algorithm 1, that is,

$$\mathcal{O}(dn_t\mathcal{C}_{\text{sys}} + dnn_t \log(n_t)).$$

As the calls to Algorithm 1 are embarrassingly parallel, the cost of Algorithm 3 reduces to

$$\mathcal{O}(\mathcal{C}_{\text{sys}} + nn_t \log(n_t)).$$

for each core if dn_t cores are available. Compared with the discussion in Sect. 2.3, this indicates that Algorithm 1 is better suited than Algorithm 2 in a massively parallel environment.

4 All-at-once Runge–Kutta formulation

In this section, we explain how the approaches presented in this work extend when the time discretization is performed by implicit Runge–Kutta methods. For this purpose, let us consider the differential problem (3) and assume, without loss of generality, that $M = I$. The discretization by an Runge–Kutta method with s stages yields

$$\begin{cases} \mathbf{x}_{j+1} = \mathbf{x}_j + \Delta t \sum_{i=1}^s b_i \mathbf{k}_i^{(j)}, & j = 0, \ldots n_t - 1, \\ \mathbf{k}_i^{(j)} = A\mathbf{x}_j + \Delta t \sum_{h=1}^s g_{ih} A\mathbf{k}_h^{(j)} + \mathbf{f}(t_j + c_i \Delta t), & i = 1, \ldots, s, \end{cases}$$

Table 1 Relative residual of the approximate solution computed by Algorithm 3 with different choices of the parameters ρ, d and for increasing size n of the problem

n	ρ	$d=1$	$d=2$	$d=3$	$d=4$
500	$1e+00$	$5.81e-02$	$2.94e-06$	$1.67e-10$	$8.80e-13$
	$1e-02$	$5.81e-04$	$2.94e-10$	$3.97e-12$	$3.49e-12$
	$1e-04$	$5.81e-06$	$1.96e-10$	$1.59e-10$	$1.38e-10$
	$1e-06$	$6.05e-08$	$1.18e-08$	$9.77e-09$	$8.19e-09$
	$1e-08$	$1.20e-06$	$8.59e-07$	$7.01e-07$	$6.07e-07$
	$1e-10$	$9.20e-05$	$6.63e-05$	$5.36e-05$	$4.58e-05$
	$1e-12$	$7.12e-03$	$5.08e-03$	$4.11e-03$	$3.59e-03$
1000	$1e+00$	$1.59e-01$	$6.47e-06$	$3.51e-10$	$1.61e-12$
	$1e-02$	$1.59e-03$	$6.48e-10$	$1.82e-11$	$1.59e-11$
	$1e-04$	$1.59e-05$	$1.17e-09$	$9.69e-10$	$8.31e-10$
	$1e-06$	$1.93e-07$	$7.84e-08$	$6.23e-08$	$5.53e-08$
	$1e-08$	$7.74e-06$	$5.44e-06$	$4.45e-06$	$3.83e-06$
	$1e-10$	$5.78e-04$	$4.11e-04$	$3.27e-04$	$2.84e-04$
	$1e-12$	$4.50e-02$	$3.16e-02$	$2.63e-02$	$2.19e-02$
2000	$1e+00$	$1.70e-01$	$6.70e-06$	$3.55e-10$	$7.06e-12$
	$1e-02$	$1.70e-03$	$6.75e-10$	$6.66e-11$	$5.73e-11$
	$1e-04$	$1.70e-05$	$3.79e-09$	$3.07e-09$	$2.60e-09$
	$1e-06$	$3.62e-07$	$2.25e-07$	$1.86e-07$	$1.58e-07$
	$1e-08$	$2.15e-05$	$1.52e-05$	$1.25e-05$	$1.08e-05$
	$1e-10$	$1.58e-03$	$1.12e-03$	$9.11e-04$	$7.96e-04$
	$1e-12$	$1.25e-01$	$9.00e-02$	$7.09e-02$	$6.38e-02$
4000	$1e+00$	$2.14e-01$	$9.97e-06$	$5.23e-10$	$3.96e-11$
	$1e-02$	$2.14e-03$	$1.04e-09$	$2.43e-10$	$2.12e-10$
	$1e-04$	$2.14e-05$	$1.46e-08$	$1.18e-08$	$1.03e-08$
	$1e-06$	$1.34e-06$	$9.35e-07$	$7.65e-07$	$6.63e-07$
	$1e-08$	$9.56e-05$	$6.73e-05$	$5.51e-05$	$4.78e-05$
	$1e-10$	$7.32e-03$	$5.15e-03$	$4.21e-03$	$3.66e-03$
	$1e-12$	$5.98e-01$	$4.23e-01$	$3.39e-01$	$2.98e-01$

where the coefficients b_i and c_i are the nodes and the weights of the Butcher tableau. By considering the Runge–Kutta matrix $G = (g_{ih}) \in \mathbb{R}^{s \times s}$ and setting $K^{(j)} := [\mathbf{k}_1^{(j)}|\ldots|\mathbf{k}_s^{(j)}] \in \mathbb{R}^{n \times s}$, we can rewrite the second equation as

$$\begin{aligned} \mathbf{k}_i^{(j)} &= A\mathbf{x}_j + \Delta t A K^{(j)}(G\mathbf{e}_i)^T + \mathbf{f}(t_j + c_i \Delta t) \\ &\Rightarrow K^{(j)} = A\mathbf{x}_j \mathbf{e}^T + \Delta t A K^{(j)} G^T + F^{(j)}, \end{aligned} \tag{24}$$

where $\mathbf{e} \in \mathbb{R}^s$ is the vector of all ones, $\mathbf{e}_i \in \mathbb{R}^s$ is the ith unit vector, and $F^{(j)} := [f(t_j + c_1 \Delta t)|\ldots|f(t_j + c_s \Delta t)] \in \mathbb{R}^{n \times s}$. Vectorizing (24) leads to

$$\underbrace{(I - \Delta t G \otimes A)}_{H} \operatorname{vec}(K^{(j)}) - (\mathbf{e} \otimes A)\mathbf{x}_j = \operatorname{vec}(F^{(j)}).$$

Therefore, we get an all-at-once linear system $\mathcal{A}\mathbf{x} = \mathbf{f}$ of the form

$$\begin{bmatrix} H & & & & & & \\ -\mathbf{b} \otimes I & I & & & & & \\ & -\mathbf{e} \otimes A & H & & & & \\ & -I & -\mathbf{b} \otimes I & I & & & \\ & & & \ddots & \ddots & & \\ & & & & -\mathbf{e} \otimes A & H & \\ & & & & -I & -\mathbf{b} \otimes I & I \end{bmatrix} \begin{bmatrix} \mathbf{k}_1^{(0)} \\ \vdots \\ \mathbf{k}_s^{(0)} \\ \mathbf{x}_1 \\ \mathbf{k}_1^{(1)} \\ \vdots \\ \mathbf{k}_s^{(1)} \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{n_t} \end{bmatrix} = \begin{bmatrix} f(t_0 + c_1 \Delta t) + A\mathbf{x}_0 \\ \vdots \\ f(t_0 + c_s \Delta t) + A\mathbf{x}_0 \\ \mathbf{x}_0 \\ f(t_1 + c_1 \Delta t) \\ \vdots \\ f(t_1 + c_s \Delta t) \\ 0 \\ f(t_2 + c_1 \Delta t) \\ \vdots \\ f(t_{n_t - 1} + c_s \Delta t) \\ 0 \end{bmatrix} \tag{25}$$

where $\mathbf{b} := \Delta t[b_1, \ldots, b_s]$ is a row vector. Note that we can rewrite the 2×2 block matrix on the main diagonal as

$$\begin{bmatrix} H & \\ -\mathbf{b} \otimes I & I \end{bmatrix} = \begin{bmatrix} -\Delta t G & \\ & \end{bmatrix} \otimes A + \begin{bmatrix} I & \\ -\mathbf{b} & 1 \end{bmatrix} \otimes I \in \mathbb{R}^{n(s+1) \times n(s+1)}.$$

Finally, by letting $\widehat{M} = \left[\begin{smallmatrix} e \otimes A \\ I \end{smallmatrix}\right] \in \mathbb{R}^{n(s+1) \times n(s+1)}$, we can write the coefficient matrix of the linear system (25) as $\mathcal{A} = I \otimes \widehat{A} + B_1 \otimes \widehat{M}$ where

$$B_1 = \begin{bmatrix} 1 & & & \\ -1 & \ddots & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \qquad \widehat{A} = \begin{bmatrix} -\Delta t G & -\mathbf{e} \\ & \end{bmatrix} \otimes A + \begin{bmatrix} I & \\ -\mathbf{b} & \end{bmatrix} \otimes I. \tag{26}$$

This implies that (25) is equivalent to a generalized Sylvester equation. In view of ParaDiag techniques, it is natural to consider the matrix $\mathcal{A}^{(\alpha)}$, obtained by replacing B_1 in (26) with an α-cyclic matrix $C_1^{(\alpha)}$. This leads to the following three extensions of the ParaDiag framework to Runge–Kutta methods.

Preconditioned Krylov method. Employ $\mathcal{A}^{(\alpha)}$ as a preconditioner for GMRES.

Evaluation interpolation approach. Employ a solver for linear systems with matrices $\mathcal{A}^{(\rho\omega_j)}$ as building block for Algorithm 3.

Low-rank updates. Solve the linear system $\mathcal{A}^{(1)}\mathbf{x} = \mathbf{f}$ and compute the solution of the corresponding update equation, that has right-hand side of rank 1, with RKSM.

In the next sections we describe the details of the solver for linear systems with the matrix $\mathcal{A}^{(\alpha)}$ and of the update equation that has to be addressed with RKSM.

4.1 Solving linear systems with $\mathcal{A}^{(\alpha)}$

After the diagonalization of the α-cyclic matrix $C_1^{(\alpha)}$, solving linear systems with the matrix $\mathcal{A}^{(\alpha)}$ requires to solve — possibly in parallel — n_t linear systems with a coefficient matrix of the form

$$\lambda_j \widehat{M} + \widehat{A} = \begin{bmatrix} -\Delta t G & (\lambda_j - 1)\mathbf{e} \end{bmatrix} \otimes A + \begin{bmatrix} I & \\ -\mathbf{b} & \lambda_j \end{bmatrix} \otimes I.$$

Each of this linear systems corresponds to solving a (generalized) Sylvester equation of the form

$$AXN_1 + XN_2 = F, \tag{27}$$

with $N_i \in \mathbb{C}^{(s+1)\times(s+1)}$, $i = 1, 2$. Because s, the number of Runge–Kutta stages, is usually small we can solve (27) efficiently by computing generalized Schur decomposition [24] of (N_1, N_2) via the QZ algorithm [29]. This yields unitary matrices $Q, Z \in \mathbb{C}^{(s+1)\times(s+1)}$ such that

$$N_1 = QT_1Z^*, \qquad N_2 = QT_2Z^*,$$

where $T_1, T_2 \in \mathbb{C}^{(s+1)\times(s+1)}$ are upper triangular. The correspondingly transformed Eq. (27) takes the form

$$AYT_1 + YT_2 = \widetilde{F}, \qquad \widetilde{F} = FZ, \qquad Y = XQ,$$

which can be solved by forward substitution by rewriting the Eq. in block-wise form:

$$A \begin{bmatrix} Y_1 & Y_2 \end{bmatrix} \begin{bmatrix} T_1^{(11)} & T_1^{(12)} \\ & T_1^{(22)} \end{bmatrix} + \begin{bmatrix} Y_1 & Y_2 \end{bmatrix} \begin{bmatrix} T_2^{(11)} & T_2^{(12)} \\ & T_2^{(22)} \end{bmatrix} = \begin{bmatrix} \widetilde{F}_1 & \widetilde{F}_2 \end{bmatrix}$$
$$\implies \begin{cases} AY_1T_1^{(11)} + Y_1T_2^{(11)} = \widetilde{F}_1 \\ AY_2T_1^{(22)} + Y_2T_2^{(22)} = \widetilde{F}_2 - AY_1T_1^{(12)} - Y_1T_2^{(12)} \end{cases}.$$

Assuming that $T_1^{(11)}, T_2^{(11)}$ are scalar quantities we can first retrieve $Y_1 \in \mathbb{C}^n$ by solving a linear system with $(A + T_2^{(11)})I)$ and continuing to compute $Y_2 \in \mathbb{C}^{n\times s}$ via recursion. The solution X of (27) is obtained as YQ^*.

4.2 Solving the update equation

In the case of Runge–Kutta methods, the low-rank update approach from Sect. 2 requires us to solve the generalized matrix equation

$$\widehat{A}\delta X + \widehat{M}\delta X B_1^T = \widehat{M} X_0 \mathbf{e}_{n_t} \mathbf{e}_1^T, \tag{28}$$

where $X_0 \in \mathbb{C}^{n(s+1)\times n_t}$ is the (reshaped) solution of the linear system $\mathcal{A}^{(1)}\mathbf{x} = \mathbf{f}$ described in the previous section. Equation (28) has a right-hand side of rank 1 but it is not immediate to recast it as a Sylvester equation of the form (9) because the coefficient $\widehat{M}$ is not invertible. Given a shift parameter $\sigma \in \mathbb{C}$, we add and subtract the quantity $\sigma\widehat{A}\delta X B_1^T$ to (28), obtaining

$$\widehat{A}\delta X(I - \sigma B_1^T) + (\widehat{M} + \sigma\widehat{A})\delta X B_1^T = \widehat{M} X_0 \mathbf{e}_{n_t}\mathbf{e}_1^T.$$

Hence, if σ is chosen such that both $\widehat{M} + \sigma\widehat{A}$ and $I - \sigma B_1^T$ are invertible, we can write an equation of the form $\widetilde{A}\delta X + \delta X\widetilde{B}^T = UV^*$ equivalent to (28) by setting:

$$\begin{aligned} \widetilde{A} &= (\widehat{M} + \sigma\widehat{A})^{-1}\widehat{A}, & \widetilde{B} &= (I - \sigma B_1)^{-1}B_1, \\ \mathbf{u} &= (\widehat{M} + \sigma\widehat{A})^{-1}\widehat{M} X_0 \mathbf{e}_{n_t}, & \mathbf{v} &= (I - \sigma B_1)^{-1}\mathbf{e}_1. \end{aligned}$$

This suggests to use the matrices $\widetilde{A}$, $\widetilde{B}$ and the vectors $\mathbf{u}$, $\mathbf{v}$ to generate Krylov subspaces for the Eq. (28). Once again, there is no need to form explicitly $\widetilde{A}$, $\widetilde{B}$ for performing matrix–vector products and solving shifted linear systems; for these operations we rely on the relations:

$$\begin{aligned} \left[(\widehat{M} + \sigma\widehat{A})^{-1}\widehat{A} - zI\right]^{-1}\mathbf{w} &= \left[(1 - z\sigma)\widehat{A} - z\widehat{M}\right]^{-1}(\widehat{M} + \sigma\widehat{A})\mathbf{w}, \\ \left[(I - \sigma B_1)^{-1}B_1 - zI\right]^{-1}\mathbf{w} &= \left[(1 + z\sigma)B_1 - zI\right]^{-1}(I - \sigma B_1)\mathbf{w}, \end{aligned}$$

that only need to manipulate the matrices $\widehat{A}$, $\widehat{M}$, B_1, $I - \sigma B_1$, which are sparse when A is sparse.

In order to study the low-rank property of the solution to (28) and to provide suitable a priori choices for the shift parameters in RKSM, one would need to study the spectral properties of $\widetilde{A}$, $\widetilde{B}$. The matrix $\widetilde{B}$ is Toeplitz triangular and we can explicitly compute its entries:

$$\widetilde{B} = (1-\sigma)^{-1}\begin{bmatrix} 1 & & & & \\ \frac{1}{\sigma-1} & \ddots & & & \\ \frac{\sigma}{(\sigma-1)^2} & \ddots & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \\ \frac{\sigma^{n_t-2}}{(\sigma-1)^{n_t-1}} & \cdots & \frac{\sigma}{(\sigma-1)^2} & \frac{1}{\sigma-1} & 1 \end{bmatrix}$$

In particular, it has the only eigenvalue $(1-\sigma)^{-1}$ with multiplicity n_t. However, the matrix $\widetilde{A}$ appears hard to analyze in general, even assuming strong properties such as A symmetric positive definite. For this reason, we postpone the analysis of (28) to future work and we propose the use of the extended Krylov method (see Example 2) or adaptive choices of the shift parameters [13] when solving the update equation with RKSM.

5 Numerical results

In this section, we compare the performances of Algorithm 2 and Algorithm 3 with those of the preconditioned GMRES method considered in [20, 37], on different examples. More specifically, we consider the following list of methods:

- `PGMRES`: GMRES preconditioned by the solver of the matrix equation with a α-circulant coefficient. If not stated otherwise, $\alpha = 1$ and the threshold for stopping GMRES is set to 10^{-8},
- `Ev-Int`: Algorithm 3. If not stated otherwise, we choose $d = 2$ and $\rho = 5 \cdot 10^{-4}$.
- `2DISCS`, `ZOL-DI(4)`, `EDS`, `EK`: Algorithm 2 with different choices of poles; see Example 1 for details. Note that, the method `ZOL-DI` is not competitive, in terms of computational time, as it requires to recompute the bases of the Krylov subspaces from scratch at every iteration. For this reason we consider its variant `ZOL-DI(4)`, that consists in cyclically repeating the 4 shifts corresponding to the Zolotarev problem with rational functions of degree (4, 4). We have chosen 4 shifts because preliminary experiments indicated that this choice offers good performance across a broader range of examples.

RKSM called at line 4 of Algorithm 2 is stopped when the relative residual of the update Eq. (9) is below the threshold 10^{-8}. The methods `EK`, `2DISCS`, and `ZOL-DI(4)` rely on precomputed factorizations of the, possibly shifted, matrices $\widetilde{A}$ and $\widetilde{B}$ when generating the bases of the rational Krylov subspaces. If the matrices A and M in (1) are sparse, a sparse direct solver is exploited whenever the solution of a linear system involving a linear combination of A and M is required; e.g., at line 4 of Algorithm 3.

For each method we report the computational time required and the relative error $\text{Res} := \|A\widehat{X}B_2^T + M\widehat{X}B_1^T - F\|_F / \|F\|_F$ in order to assess the accuracy of the correspondent approximate solution $\widehat{X}$. The best timings of each case study are highlighted in bold and reported in Tables 2, 3, 4, 5, 6, 7, 8. For the various implementations of Algorithm 2 we also report the percentage of the CPU time that is spent for solving (9), labeled with T_{sylv}, the dimension (Dim) of the rational Krylov subspace generated and the rank (rk) of the approximate solution to (9), after recompression.

Apart from Sect. 5.7, all experiments have been performed on a laptop with a dual-core Intel Core i7-7500U 2.70 GHz CPU, 256KB of level 2 cache, and 16 GB of RAM. The algorithms are implemented in MATLAB and tested under MATLAB2020a, with MKL BLAS version 2019.0.3 utilizing both cores.

5.1 1D heat equation

We start by testing all the algorithms detailed at the beginning of this section on the matrix equation $AX + XB_1^T = F$ from Example 1 for $n = 1089, 4225$ and $n_t = 2^8, \ldots, 2^{11}$.

The results are reported in Table 2. As expected `ZOL-DI(4)` and `EDS` generate significantly lower dimensional Krylov subspaces with respect to those generated by `EK` and `2DISCS`, while achieving comparable accuracies. Looking at the computational times, we see that, for all methods, we have an almost linear dependence on the size n and on the number of time steps n_t. The methods with the most favorable timings are `Ev-Int` and `ZOL-DI(4)`. The former is preferable when $\mathrm{T}_{\mathrm{sylv}} > 50\%$, which happens for the smaller instances of the equation. For all four implementations of Algorithm 2, the fraction of the time spent for solving the low-rank Sylvester equation decreases as the size of the problem increases. Since the advantage of using `ZOL-DI(4)` is inversely proportional to $\mathrm{T}_{\mathrm{sylv}}$ we get the best scenario for `ZOL-DI(4)` when $n = 4225$ and $n_t = 2048$ where we get about a 2× speedup over `Ev-Int` and a 5× speedup over `PGMRES`.

5.2 Convection diffusion

This example is concerned with the convection diffusion equation from [37, Section 6.2], which models the temperature distribution in a cavity having an external wall maintained at a constant (hot) temperature and a wind that determines a recirculating flow. The problem reads as follows:

$$\begin{cases} u_t - \epsilon\Delta u + \mathbf{w}\cdot\nabla u = 0, & S \times (0,1), \ \ S = (-1,1)^2, \\ u(x,y,0) = \chi_{\{x=1\}}, & \\ u(x,y,t) = \chi_{\{x=1\}}, & \partial S \times (0,1), \end{cases}$$

where $\epsilon = 0.005$, $\chi_{\{x=1\}}$ is the indicator function of the set $\{x = 1\}$, and $\mathbf{w} = (2y(1-x^2), -2x(1-y^2))$. The discretization is done via Q1 finite elements over the domain S and backward Euler time-stepping, together with streamline-upwind Petrov–Galerkin (SUPG) stabilization. The matrix equation corresponding to the all-at-once linear system takes the form $AX + MXB_1^T = F$, where both A and M are sparse. To estimate the numerical range of $M^{-1}A$, we compute the largest and smallest real parts, $r_{\max}, r_{\min}$, of its eigenvalues, by means of the Matlab command `eigs`. We remark that, $r_{\max}, r_{\min} > 0$ and we denote $c := (r_{\max} + r_{\min})/2$. Then, `2DISCS` employs $\{z : \ |z - c| \le (r_{\max} - r_{\min})/2\}$ as estimate for $\mathcal{W}(M^{-1}A)$ while `ZOL-DI(4)` and `EDS` makes use of the real interval $[r_{\min}, r_{\max}]$. The performances of the numerical methods are reported in Table 3. Although we used the slightly higher tolerance 10^{-6} to stop `PGMRES`, the latter needs 17 iterations to converge and this makes the other approaches significantly faster. Among the methods based on low-rank updates, `ZOL-DI(4)` and `EDS` are those that generate the Krylov subspaces of smallest dimensions, while maintaining a good accuracy. This feature makes `EDS` faster than `EK` although the latter has a cheaper iteration cost. Similar comments to the ones made in the previous example apply to the comparison between `ZOL-DI(4)` and `Ev-Int`.

Table 2 1D heat equation; performances of the various methods

n	n_t	PGMRES			Ev-Int		EK					2DISCS				
		Time	Res	It	Time	Res	Time	Res	rk	T_{sylv} (%)	Dim	Time	Res	rk	T_{sylv} (%)	Dim
1089	256	0.25	$7.2e-10$	3	**0.10**	$2.4e-09$	0.39	$1.4e-08$	12	88.3	47	0.30	$8.9e-09$	12	85.1	49
	512	0.47	$2.1e-09$	3	**0.21**	$2.3e-09$	0.48	$1.0e-08$	13	78.0	59	0.57	$5.1e-09$	13	81.6	67
	1024	0.98	$1.6e-09$	3	0.40	$2.4e-09$	0.70	$2.5e-08$	14	75.4	71	1.00	$2.1e-08$	14	82.7	89
	2048	2.02	$3.5e-09$	3	0.82	$2.3e-09$	1.14	$3.3e-08$	15	68.3	89	1.91	$2.8e-08$	15	80.9	121
4225	256	1.08	$2.9e-08$	3	0.40	$3.4e-08$	0.54	$1.4e-08$	13	70.2	47	0.63	$9.0e-09$	13	74.4	49
	512	2.27	$3.0e-08$	3	0.75	$3.4e-08$	0.86	$1.0e-08$	14	62.4	59	1.08	$5.2e-09$	14	69.5	67
	1024	4.56	$1.3e-07$	3	1.69	$3.7e-08$	1.66	$2.5e-08$	15	52.0	71	2.21	$2.1e-08$	15	63.6	89
	2048	9.72	$3.1e-07$	3	3.28	$3.6e-08$	2.90	$3.3e-08$	17	48.3	89	4.07	$2.9e-08$	17	62.6	121

n	n_t	EDS					ZOL-DI(4)				
		Time	Res	rk	T_{sylv} (%)	Dim	Time	Res	rk	T_{sylv} (%)	Dim
1089	256	0.24	$2.6e-10$	12	81.3	25	0.15	$4.7e-11$	12	69.7	33
	512	0.27	$7.8e-09$	13	61.4	25	0.22	$7.8e-11$	13	51.9	33
	1024	0.37	$1.6e-08$	14	53.7	29	**0.33**	$3.0e-09$	14	47.4	33
	2048	0.61	$9.7e-09$	15	39.9	32	**0.51**	$5.1e-09$	15	28.6	37
4225	256	0.41	$7.2e-10$	13	60.2	25	**0.33**	$6.8e-10$	13	50.9	33
	512	0.59	$7.9e-09$	14	43.4	25	**0.51**	$7.1e-10$	14	34.7	33
	1024	1.25	$1.6e-08$	15	36.3	29	**0.96**	$3.1e-09$	15	17.4	33
	2048	1.85	$9.7e-09$	17	18.4	32	**1.79**	$5.0e-09$	17	15.7	37

Table 3 Convection diffusion; performances of the various methods

n	n_t	PGMRES			Ev-Int		EK					2DISCS				
		Time	Res	It	Time	Res	Time	Res	rk	T_{sylv} (%)	Dim	Time	Res	rk	T_{sylv} (%)	Dim
1089	256	19.66	$4.2e-08$	17	2.05	$7.2e-10$	3.45	$2.3e-10$	28	70.6	165	2.66	$2.2e-10$	28	61.9	123
	512	42.25	$7.0e-08$	17	4.29	$1.0e-09$	4.90	$4.4e-11$	29	70.2	171	4.22	$5.3e-11$	29	52.0	141
	1024	81.87	$1.0e-07$	17	8.62	$1.5e-09$	6.79	$1.7e-10$	30	55.7	165	6.79	$2.5e-10$	30	48.5	153
	2048	168.41	$1.4e-07$	17	17.00	$2.1e-09$	11.20	$4.7e-11$	31	55.0	165	11.20	$6.4e-11$	31	40.9	159
4225	256	92.15	$1.7e-08$	17	9.59	$8.4e-11$	15.95	$4.7e-10$	31	36.9	251	9.90	$3.4e-10$	32	37.0	147
	512	191.68	$2.6e-08$	17	18.45	$1.2e-10$	20.54	$7.1e-11$	33	38.3	259	15.65	$7.2e-11$	33	29.0	171
	1024	367.04	$3.9e-08$	17	36.18	$1.7e-10$	29.36	$5.4e-10$	34	24.6	259	25.52	$6.4e-10$	35	24.6	189
	2048	725.21	$5.6e-08$	17	72.69	$2.4e-10$	47.54	$1.7e-10$	35	23.3	251	44.69	$1.1e-10$	35	18.4	197

n	n_t	EDS					ZOL-DI(4)				
		Time	Res	rk	T_{sylv} (%)	Dim	Time	Res	rk	T_{sylv} (%)	Dim
1089	256	2.03	$1.6e-10$	28	50.0	67	**1.49**	$1.8e-10$	28	32.0	65
	512	3.13	$2.0e-11$	29	41.6	70	**2.72**	$2.3e-11$	29	30.8	69
	1024	5.29	$2.7e-10$	30	30.8	72	**4.84**	$1.8e-10$	30	20.1	69
	2048	9.57	$8.7e-11$	31	25.9	75	**9.11**	$3.6e-11$	31	18.9	77
4225	256	8.13	$3.1e-10$	31	19.2	82	**6.87**	$3.0e-10$	31	11.6	81
	512	12.48	$6.5e-11$	33	16.5	84	**11.39**	$3.0e-11$	33	10.9	85
	1024	21.70	$3.6e-10$	35	11.7	92	**20.34**	$1.9e-10$	35	7.2	89
	2048	40.20	$1.4e-10$	35	9.3	93	**38.75**	$7.1e-11$	35	5.9	89

5.3 Fractional space diffusion

We consider the fractional diffusion example from [51, Section 5]:

$$\begin{cases} u_t = a_1 \frac{\partial^{\gamma_1} u^+}{\partial x^{\gamma_1}} + b_1 \frac{\partial^{\gamma_1} u^-}{\partial x^{\gamma_1}} + a_2 \frac{\partial^{\gamma_2} u^+}{\partial y^{\gamma_2}} + b_2 \frac{\partial^{\gamma_2} u^-}{\partial y^{\gamma_2}} + f(x, y, t), & S \times (0, 5), \ \ S = (0, 1)^2, \\ u(x, y, 0) = 0, & (x, y) \in S, \\ u(x, y, t) = 0, & (x, y, t) \in \partial S \times (0, 5), \end{cases}$$

where $\frac{\partial^{\gamma_j} u^{\pm}}{\partial x}$ indicates the Riemann-Liouville right- and left- looking derivatives; the diffusion coefficients are set as $a_1 = 1, a_2 = 0.5, b_1 = 0.2, b_2 = 1$, the differential orders are $\gamma_1 = 1.75$ and $\gamma_2 = 1.5$, and the source term is $f(x, y, t) = 10 \sin(3xyt)$. For space discretization, the second-order weighted and shifted Grünwald difference formula is employed; uniform backward Euler time stepping is applied. The all-at-once linear system matrix takes the form $I \otimes A + B_1 \otimes I$ where the matrix A has the structure

$$A = I \otimes T_1 + T_2 \otimes I$$

with T_1, T_2 non sparse Toeplitz matrices; see [51] for a detailed description. In particular, matrix–vector operations with the matrix A are efficiently performed by relying on FFT and matrix equation techniques. Moreover, the minimum and maximum real part of the eigenvalues of A are easily obtained once estimates of the same quantities for the matrices T_1, T_2 are computed. When running `2DISCS`, `ZOL-DI(4)`, `EDS`, we approximate $\mathcal{W}(A)$ as we did in the previous example.

The results reported in Table 4 show that the methods based on low-rank updates struggle on this example because they have to generate somewhat large subspaces in order to achieve the desired accuracy. This is caused by the fact that the solution of the update Eq. (8) has a slightly higher rank with respect to the previous examples, and the approximations of $\mathcal{W}(A)$ are not tight, see Fig. 4. Nevertheless, `ZOL-DI(4)` achieves the best timings for $n = 4225$. `Ev-Int` maintains the usual behavior and turns out to be the preferable method for $n = 1089$.

5.4 Fractional time diffusion

We now consider an example where Algorithm 2 is not applicable due to the non locality of the time differential operator. We modify Example 2 by replacing the first-order time derivative with the Caputo time derivative of order $\gamma \in (0, 1)$:

Table 4 Fractional space diffusion; performances of the various methods

n	n_t	PGMRES			Ev-Int		EK					2DISCS				
		Time	Res	It	Time	Res	Time	Res	rk	T_{sylv} (%)	Dim	Time	Res	rk	T_{sylv} (%)	Dim
1089	256	**2.03**	$3.9e-12$	2	0.92	$4.9e-11$	2.35	$1.3e-07$	26	74.4	113	1.45	$3.8e-08$	26	58.5	97
	512	**3.42**	$1.3e-10$	2	1.68	$8.1e-11$	4.36	$1.6e-07$	33	79.4	171	2.41	$5.8e-08$	33	62.7	133
	1024	**8.74**	$1.2e-10$	3	3.69	$1.0e-10$	13.15	$1.7e-07$	40	86.4	265	4.69	$2.5e-07$	40	62.0	169
	2048	**21.64**	$1.2e-11$	4	7.14	$5.8e-08$	29.90	$3.3e-07$	46	87.8	355	7.59	$3.9e-07$	46	52.0	197
4225	256	7.30	$3.5e-11$	2	3.58	$4.2e-10$	3.78	$1.0e-07$	25	51.6	107	3.48	$1.2e-07$	25	47.4	97
	512	14.47	$7.2e-11$	2	7.15	$5.5e-10$	10.01	$2.4e-07$	32	63.8	179	6.74	$1.1e-07$	32	46.1	141
	1024	35.71	$1.9e-08$	2	14.16	$4.3e-10$	22.50	$3.0e-07$	42	68.7	255	13.86	$2.4e-07$	42	49.2	197
	2048	88.22	$1.0e-08$	3	29.11	$1.6e-09$	54.76	$5.9e-07$	55	71.8	375	26.86	$4.4e-07$	55	42.4	265

n	n_t	EDS					ZOL-DI(4)				
		Time	Res	rk	T_{sylv} (%)	Dim	Time	Res	rk	T_{sylv} (%)	Dim
1089	256	2.37	$1.3e-07$	26	74.7	88	1.34	$8.9e-08$	26	53.8	85
	512	3.96	$1.9e-07$	33	77.3	121	2.14	$1.6e-07$	33	57.9	117
	1024	9.24	$2.8e-07$	40	80.7	175	4.54	$1.4e-07$	40	60.7	169
	2048	16.85	$1.9e-07$	46	78.4	220	8.61	$3.9e-07$	46	57.6	213
4225	256	4.73	$2.1e-07$	25	61.3	89	**3.09**	$3.1e-08$	25	40.7	89
	512	9.42	$2.6e-07$	32	61.5	126	**6.23**	$3.0e-07$	32	41.6	121
	1024	16.70	$2.8e-07$	42	57.9	170	**11.23**	$2.9e-07$	42	37.3	165
	2048	41.61	$5.5e-07$	55	62.8	256	**24.59**	$5.1e-07$	55	37.1	237

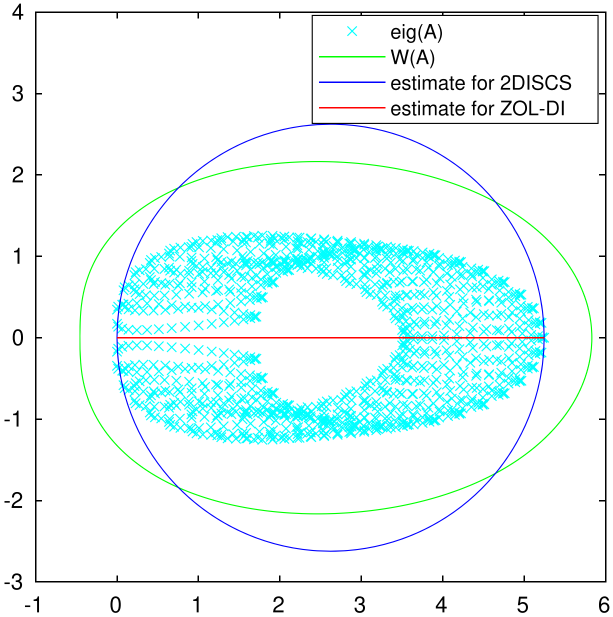

Fig. 4 Fractional diffusion example with $n = 1089$; eigenvalues of the matrix A (cyan crosses), numerical range of the matrix A (disc enclosed by the green line) and its approximations used for `2DISCS` (disc enclosed by the blue line), and for `ZOL-DI(4)` (red line segment)

$$\begin{cases} \frac{\partial^\gamma u}{\partial t^\gamma} = u_{xx} + f(x,t), & S := [0,1] \times [0,1], \\ u \equiv 0, & \text{on } \partial S, \\ u(x,0) = 4x(1-x), & \text{at } t = 0, \\ f(x,t) = h \max\{1 - |c(t) - x|/w, 0\}, & c(t) = \frac{1}{2} + (\frac{1}{2} - w)\sin(2\pi t), \end{cases}$$

and we keep the same parameters w, h of the integer derivative case. Discretizing the Caputo time derivative with the unshifted Grünwald-Letnikov formula, we get a matrix equation $AX + XB_1^T = F$ where B_1 is Toeplitz lower triangular and takes the form:

$$B_1 = \begin{bmatrix} g_{\gamma,0} & & & \\ g_{\gamma,1} & g_{\gamma,0} & & \\ \vdots & \ddots & \ddots & \\ g_{\gamma,n_t-1} & \cdots & g_{\gamma,1} & g_{\gamma,0} \end{bmatrix}, \qquad \begin{cases} g_{\gamma,0} = 1, \\ g_{\gamma,j} = \left(1 - \frac{\gamma+1}{j}\right) g_{\gamma,j-1}. \end{cases}$$

In particular, the usual way to construct a circulant matrix from B_1 requires to apply a matrix $\delta B_1^{(\alpha)}$ of rank $n_t - 1$. For this reason we restrict to `PGMRES` and `Ev-Int` for solving the all-at-once problem. The results reported in Table 5 refer to the case $\gamma = 0.3$ and show that `Ev-Int` significantly outperform `PGMRES`, despite the low number of iterations needed by the latter to converge.

Table 5 Fractional time differentiation; performances of PGMRES and Ev-Int

n	n_t	PGMRES			Ev-Int	
		Time	Res	It	Time	Res
1089	256	0.46	$1.0e-06$	5	**0.12**	$1.6e-09$
	512	0.74	$7.8e-05$	4	**0.22**	$1.3e-09$
	1024	1.55	$2.7e-05$	4	**0.45**	$1.2e-09$
	2048	3.68	$8.0e-06$	4	**0.87**	$9.9e-10$
4225	256	1.56	$1.6e-05$	5	**0.39**	$2.3e-08$
	512	2.77	$1.2e-03$	4	**0.87**	$2.0e-08$
	1024	6.45	$4.1e-04$	4	**1.66**	$1.9e-08$
	2048	14.41	$1.2e-04$	4	**3.71**	$1.5e-08$

5.5 Wave equation

We consider the linear wave equation

$$\begin{cases} u_{tt} - \Delta u = (1+2\pi^2)e^t \sin(\pi x)\sin(\pi y), & \text{on } S \times (0,T), \\ u(x,y,t) = 0, & \text{on } \partial S \times (0,T), \\ u(\cdot,\cdot,0) = u_0(x,y), \quad u_t(\cdot,\cdot,0) = u_1(x,y), & \text{in } S, \end{cases}$$

with $S = (0,1)$, $T = 1$, $u_0(x,y) = u_1(x,y) = \sin(\pi x)\sin(\pi y)$. As described in [20, Section 3], by discretizing with finite differences in space and with the implicit leap-frog scheme in time we get the all-at-once linear system $(B_1 \otimes I + B_2 \otimes A)\mathbf{x} = \mathbf{f}$ where $A = \text{trid}(-1,2,-1)\cdot(n+1)^2$ and

$$B_1 = \frac{1}{\Delta t^2}\begin{bmatrix} 1 & & & & \\ -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \end{bmatrix} \in \mathbb{R}^{n_t \times n_t}, \qquad B_2 = \frac{1}{2}\begin{bmatrix} 1 & & & & \\ 0 & 1 & & & \\ 1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{n_t \times n_t}.$$

We note that, for this example, all methods based on low-rank updates generate quite small Krylov subspaces, and their dimensions do not grow as the parameters n and n_t vary. As a consequence, EK is the one that achieves the cheapest consumption of computational time and we only report its performances. We remark that, in order to get comparable accuracy we have considered $d = 3$ roots of the unit when running Ev-Int. The preconditioner employed in PGMRES uses the parameter $\alpha = 0.1$ as suggested in [20] and the tolerance for stopping the GMRES iteration has been set to 10^{-7}. The results are shown in Table 6. Comparing PGMRES, Ev-Int, and EK, leads to similar conclusions as in the example of the 1D heat equation.

Table 6 Wave equation; performances of the various methods

n	n_t	PGMRES			Ev-Int		EK				
		Time	Res	It	Time	Res	Time	Res	rk	T_{sylv} (%)	Dim
1089	257	0.49	$1.2e-12$	3	**0.20**	$2.7e-10$	0.50	$2.9e-10$	26	86.4	27
	513	0.72	$3.1e-13$	3	**0.34**	$2.6e-10$	0.53	$1.3e-10$	26	77.5	27
	1025	1.42	$3.9e-12$	3	**0.71**	$2.5e-10$	**0.71**	$5.3e-11$	27	67.1	27
	2049	3.34	$1.7e-12$	3	1.50	$2.5e-10$	**1.25**	$3.7e-11$	27	61.7	27
4225	257	1.33	$3.7e-12$	3	0.64	$2.8e-10$	**0.57**	$2.9e-10$	26	55.5	27
	513	2.65	$7.3e-12$	3	1.27	$2.7e-10$	**0.76**	$1.3e-10$	26	54.0	27
	1025	5.96	$4.6e-11$	3	2.96	$2.6e-10$	**1.25**	$5.4e-11$	26	34.5	27
	2049	12.00	$3.2e-11$	3	5.38	$2.5e-10$	**2.65**	$3.5e-11$	27	34.1	27

5.6 Runge–Kutta example

We present a numerical test on the all-at-once Runge–Kutta formulation introduced in Sect. 4. Let us start from the finite difference semidiscretization of the following differential problem:

$$\begin{cases} u_t = u_{xx}, & (x,t) \in (0,1)^2, \\ u(x,0) = \sin(\pi x), \end{cases}$$
$$\Rightarrow \quad \begin{cases} \dot{U}(t) = AU(t), & A = \operatorname{trid}(1,-2,1)\cdot(n+1)^2, \\ U(0) = \mathbf{x}_0, & (\mathbf{x}_0)_j = \sin(j\pi/(n+1)). \end{cases} \tag{29}$$

We consider solving the all-at-once formulation (25) corresponding to the following four-stage ($s = 4$), 3rd order, *Diagonally Implicit Runge-Kutta method* (DIRK):

$$\begin{array}{c|cccc} 1/2 & 1/2 & & & \\ 2/3 & 1/6 & 1/2 & & \\ 1/2 & -1/2 & 1/2 & 1/2 & \\ 1 & 3/2 & -3/2 & 1/2 & 1/2 \\ \hline & 3/2 & -3/2 & 1/2 & 1/2 \end{array} .$$

More specifically, we solve (25) by applying the methods PGMRES, Ev-Int and EK adapted as described in Sect. 4. For this experiment, the parameter ρ used by Ev-Int is set to the value 0.05. As shift parameter for the update Eq. (28) solved by EK, it has been chosen $\sigma = -2$. The performances of the 3 methods are reported in Table 7; as observed in previous tests, the approaches based on low-rank update and evaluation interpolation outperforms PGMRES for both speed and accuracy. As the number of time steps increases, the cost of the update equation becomes relatively cheap, making EK faster than Ev-Int. The relative residuals obtained with EK and Ev-Int are comparable.

Table 7 Performances of the Runge–Kutta schemes for different values of $n = 250, 500$ and $n_t = 250, 500, 1000, 2000$

$n(s+1)$	n_t	PGMRES			Ev-Int		EK				
		Time	Res	It	Time	Res	Time	Res	rk	T_{sylv} (%)	Dim
1250	250	0.78	$7.2e-08$	3	**0.24**	$1.4e-11$	0.36	$2.1e-11$	1	69.0	27
	500	1.34	$3.2e-09$	3	**0.47**	$1.0e-11$	0.49	$1.1e-11$	1	52.7	31
	1000	3.16	$1.5e-07$	3	0.90	$1.7e-11$	**0.72**	$2.7e-12$	2	41.1	33
	2000	6.64	$2.8e-07$	3	2.05	$5.9e-11$	**1.24**	$4.0e-12$	2	25.9	33
2500	250	1.50	$1.8e-07$	3	**0.41**	$1.2e-10$	0.63	$3.3e-12$	2	68.9	27
	500	3.21	$1.6e-07$	3	**0.83**	$6.6e-11$	0.88	$4.9e-12$	2	53.5	31
	1000	6.49	$3.4e-06$	3	1.80	$6.2e-11$	**1.39**	$7.2e-12$	2	39.7	33
	2000	13.35	$7.9e-05$	3	3.73	$2.2e-10$	**2.41**	$1.1e-11$	2	24.3	33

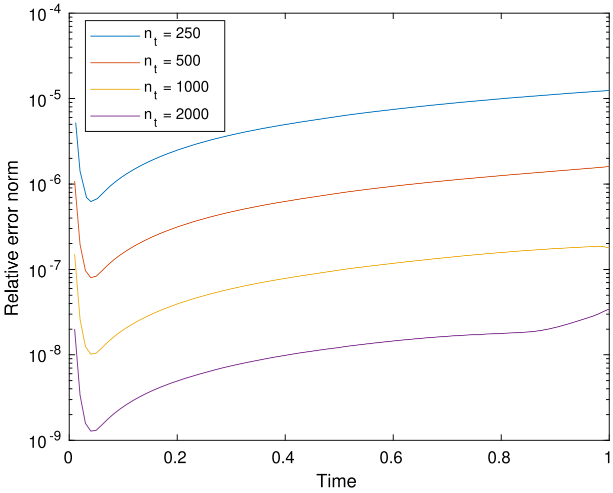

Fig. 5 Relative error norm of EK for $n = 500$ and $n_t = 250, 500, 1000, 2000$; the error decreases with order 3 with respect to the time step size

Problem (29) admits the explicit solution $U(t) = \exp(tA)\mathbf{x}_0$ so that we can compare approximate solutions with respect to a benchmark solution computed by means of the expm Matlab function. Figure 5 reports the relative error $\|\widehat{\mathbf{x}}_t - \exp(tA)\mathbf{x}_0\|_2/\|\exp(tA)\mathbf{x}_0\|_2$ of the approximants $\widehat{\mathbf{x}_t}$ returned by EK, for $n = 500$ and different time step sizes; the plot is in agreement with the order 3 of the Runge–Kutta scheme.

5.7 Preliminary experiments on parallelism

We conclude by testing the parallel scaling features of the proposed algorithms in a multi-core computational environment. The experiment in this section has been run on a server with two Intel(R) Xeon(R) E5-2650v4 CPU with 12 cores and 24

Table 8 Fractional space diffusion with $n = 4225$ and $n_t = 8192$; parallel efficiency of the procedures as the number of cores p increases

p	PGMRES		Ev-Int		ZOL-DI		
	Time	SE	Time	SE	Time	T_{sylv} (%)	SE
1	393.23	100.0	128.18	100.0	**118.27**	46.3	100.0
2	206.50	95.2	**66.91**	95.8	87.30	62.5	67.7
4	135.94	72.3	**38.13**	84.0	71.45	74.5	41.4
8	74.44	66.0	**20.66**	77.5	63.02	84.9	23.5
12	53.64	61.1	**14.91**	71.6	60.18	89.2	16.4

threads each, running at 2.20 GHz, using MATLAB R2022b with the Intel(R) Math Kernel Library Version 11.3.1. The test has been run using the SLURM scheduler, allocating 12 cores and 250 GB of RAM. Our implementation exploits parallelism for the solution of the block diagonal linear system generated by FAST_DIAG_SOLVE (lines 4-6 of Algorithm 1). This parallel routine is used as a preconditioner in PGMRES, it is called at line 2 of Algorithm 2, and it is called at line 4 of Ev-Int. Finally, we also parallelize the for loop in Ev-Int (lines 2-5 of Algorithm 3).

As case study, we consider the fractional diffusion example of Sect. 5.3 with $n = 4225$ and $n_t = 8192$, and we measure the execution time of the procedures PGMRES, Ev-Int, and ZOL-DI, as the number of cores p ranges in the set $\{1, 2, 4, 8, 12\}$. The tolerance to stop GMRES has been set to 10^{-6} in order to get comparable residual norms with the other two methods. To evaluate the efficiency of the implementation, we also compute the strong scaling efficiency (SE) with p cores defined as

$$\mathrm{SE}(p) := 100 \cdot \frac{\mathrm{Time}(1\text{ core})}{p \cdot \mathrm{Time}(p\text{ cores})}.$$

The results reported in Table 8 clearly show that Ev-Int is the method that gains the most from parallelism, maintaining more than 70% of strong scaling efficiency, up to 12 cores. Also PGMRES leverages the use of multiple cores to the extent of becoming faster than ZOL-DI, when $p = 12$. On the contrary, our current implementation of ZOL-DI does not exploit more than one core for solving the update equation and, consequently, parallel efficiency drops as soon as the cost of (8) becomes dominant. An adaptation of the parallel implementation of RKSM [8] is nontrival and beyond the scope of this work, but it potentially addresses the scaling issue of ZOL-DI.

6 Conclusions

We have proposed a tensorized Krylov subspace method and an interpolation method for replacing the iterative refinement phase that is required by ParaDiag algorithms with uniform time steps. In the case of multistep methods, a theoretical analysis of the approximation property of the Krylov subspace method has been provided. We have shown several numerical tests demonstrating that our approaches can significantly outperform block-circulant preconditioned GMRES iterations for solving linear initial value problems. Let us emphasize that the focus of this work was on theoretical and algorithmic aspects; we have not carried out an extensive study of the parallel implementation of our algorithms, which will be subject to future work. From the analysis in Sects. 2.3 and 3.1 we expect that the interpolation method (Algorithm 3) will perform significantly better in a massively parallel environment.

We remark that Algorithm 2 and Algorithm 3 can directly be put in place instead of the existing ParaDiag algorithms for solving certain non-linear ODEs of the form $M\dot{U} + f(U) = 0$ [17] and for computing the so-called *Coarse Grid Correction* of the Parareal algorithm [22, 33].

In Sect. 4, we have discussed how to extend the methodology to the case of implicit Runge–Kutta time integration, but some questions remain for further work. In particular, a careful analysis of the low-rank approximability of Eq. (28) and of the choice of the shift parameter σ, would be desirable.

Acknowledgements The authors thank Bernd Beckermann for inspiring discussions on the Zolotarev problems in Sect. 2.2.

Funding Open access funding provided by Università di Pisa within the CRUI-CARE Agreement.

References

1. Antoulas, A.C., Sorensen, D.C., Zhou, Y.: On the decay rate of Hankel singular values and related issues. Syst. Control Lett. **46**(5), 323–342 (2002)
2. Baker, J., Embree, M., Sabino, J.: Fast singular value decay for Lyapunov solutions with nonnormal coefficients. SIAM J. Matrix Anal. Appl. **36**(2), 656–668 (2015)
3. Beckermann, B.: An error analysis for rational Galerkin projection applied to the Sylvester equation. SIAM J. Numer. Anal. **49**(6), 2430–2450 (2011)
4. Beckermann, B., Townsend, A.: Bounds on the singular values of matrices with displacement structure. SIAM Rev. **61**(2), 319–344 (2019)
5. Benner, P., Li, R.-C., Truhar, N.: On the ADI method for Sylvester equations. J. Comput. Appl. Math. **233**(4), 1035–1045 (2009)
6. Benner, P., Saak, J.: Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey. GAMM-Mitt. **36**(1), 32–52 (2013)
7. Berljafa, M., Güttel, S.: Generalized rational Krylov decompositions with an application to rational approximation. SIAM J. Matrix Anal. Appl. **36**(2), 894–916 (2015)
8. Berljafa, M., Güttel, S.: Parallelization of the rational Arnoldi algorithm. SIAM J. Sci. Comput. **39**(5), S197–S221 (2017)
9. Bertaccini, D.: The spectrum of circulant-like preconditioners for some general linear multistep formulas for linear boundary value problems. SIAM J. Numer. Anal. **40**(5), 1798–1822 (2002)
10. Bertaccini, D., Ng, M.K.: Block $\{\omega\}$-circulant preconditioners for the systems of differential equations. Calcolo **40**(2), 71–90 (2003)

11. Crouzeix, M., Palencia, C.: The numerical range is a $(1+\sqrt{2})$-spectral set. SIAM J. Matrix Anal. Appl. **38**(2), 649–655 (2017)
12. Druskin, V., Knizhnerman, L., Zaslavsky, M.: Solution of large scale evolutionary problems using rational Krylov subspaces with optimized shifts. SIAM J. Sci. Comput. **31**(5), 3760–3780 (2009)
13. Druskin, V., Simoncini, V.: Adaptive rational Krylov subspaces for large-scale dynamical systems. Syst. Control Lett. **60**(8), 546–560 (2011)
14. Ellner, N.S., Wachspress, E.L.: New ADI model problem applications. In: Proceedings of 1986 ACM Fall Joint Computer Conference, pp. 528–534, (1986)
15. Gander, M.J.: 50 years of time parallel time integration. In: Multiple Shooting and Time Domain Decomposition Methods, vol. 9 of Contrib. Math. Comput. Sci., pp. 69–113. (Cham: Springer, 2015)
16. Gander, M.J., Güttel, S.: PARAEXP: a parallel integrator for linear initial-value problems. SIAM J. Sci. Comput. **35**(2), C123–C142 (2013)
17. Gander, M.J., Halpern, L.: Time parallelization for nonlinear problems based on diagonalization. In: Domain Decomposition Methods in Science and Engineering XXIII, vol. 116 of Lect. Notes Comput. Sci. Eng., pp. 163–170. (Cham: Springer, 2017)
18. Gander, M.J., Halpern, L., Rannou, J., Ryan, J.: A direct time parallel solver by diagonalization for the wave equation. SIAM J. Sci. Comput. **41**(1), A220–A245 (2019)
19. Gander, M.J., Halpern, L., Ryan, J., Tran, T.T.B.: A direct solver for time parallelization. In: Domain Decomposition Methods in Science and Engineering XXII, vol. 104 of Lect. Notes Comput. Sci. Eng., pp. 491–499. (Cham: Springer, 2016)
20. Gander, M.J., Liu, J., Wu, S.-L., Yue, X, Zhou, T.: ParaDiag parallel-in-time algorithms based on the diagonalization technique. arXiv preprint arXiv:2005.09158, (2020)
21. Gander, M.J., Lunet, T., Ruprecht, D., Speck, R.: A unified analysis framework for iterative parallel-in-time algorithms. arXiv preprint arXiv:2203.16069, (2022)
22. Gander, M.J., Vandewalle, S.: Analysis of the parareal time-parallel time-integration method. SIAM J. Sci. Comput. **29**(2), 556–578 (2007)
23. Gander, M.J., Wu, S.-L.: Convergence analysis of a periodic-like waveform relaxation method for initial-value problems via the diagonalization technique. Numer. Math. **143**(2), 489–527 (2019)
24. Golub, G.H., Van Loan, C.F.: Johns Hopkins Studies in the Mathematical Sciences, 4th edn. Johns Hopkins University Press, Baltimore (2013)
25. Gončar, A.A.: The problems of E. I. Zolotarev which are connected with rational functions. Mat. Sb. **78**(120), 640–654 (1969)
26. Grimme, E.J.: Krylov Projection Methods for Model Reduction. University of Illinois at Urbana-Champaign, Champaign (1997)
27. Hairer, E., Wanner, G.: Solving ordinary differential equations. II, vol. 14 of Springer Series in Computational Mathematics. (Berlin: Springer-Verlag 2010)
28. Henrici, P.: Applied And Computational Complex Analysis. Vol. 1. Wiley Classics Library. John Wiley & Sons, Inc., New York, 1988. Power series—integration—conformal mapping—location of zeros, Reprint of the (1974) original, A Wiley-Interscience Publication
29. Kågström, B., Kressner, D.: Multishift variants of the QZ algorithm with aggressive early deflation. SIAM J. Matrix Anal. Appl. **29**(1), 199–227 (2006)
30. Kressner, D., Massei, S., Robol, L.: Low-rank updates and a divide-and-conquer method for linear matrix equations. SIAM J. Sci. Comput. **41**(2), A848–A876 (2019)
31. Le Bailly, B., Thiran, J.P.: Optimal rational functions for the generalized Zolotarev problem in the complex plane. SIAM J. Numer. Anal. **38**(5), 1409–1424 (2000)
32. Lebedev, V.: On a Zolotarev problem in the method of alternating directions. USSR Comput. Math. Math. Phys. **17**(2), 58–76 (1977)
33. Lions, J.-L., Maday, Y., Turinici, G.: Résolution d'EDP par un schéma en temps pararéel. C. R. Acad. Sci. Paris Sér I Math. **332**(7), 661–668 (2001)
34. Liu, J., Wu, S.-L.: A fast block α-circulant preconditoner for all-at-once systems from wave equations. SIAM J. Matrix Anal. Appl. **41**(4), 1912–1943 (2020)
35. Maday, Y., Rønquist, E.M.: Parallelization in time through tensor-product space-time solvers. C. R. Math. Acad. Sci. Paris **346**(1–2), 113–118 (2008)
36. Massei, S., Robol, L.: Rational Krylov for Stieltjes matrix functions: convergence and pole selection. BIT **61**(1), 237–273 (2021)
37. McDonald, E., Pestana, J., Wathen, A.: Preconditioning and iterative solution of all-at-once systems for evolutionary partial differential equations. SIAM J. Sci. Comput. **40**(2), A1012–A1033 (2018)

38. Noschese, S., Reichel, L.: Generalized circulant Strang-type preconditioners. Numer. Linear Algebra Appl. **19**(1), 3–17 (2012)
39. Palitta, D.: Matrix equation techniques for certain evolutionary partial differential equations. J. Sci. Comput. **87**(3), 36 (2021)
40. Peaceman, D.W., Rachford, H.H., Jr.: The numerical solution of parabolic and elliptic differential equations. J. Soc. Indust. Appl. Math. **3**, 28–41 (1955)
41. Penzl, T.: Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case. Syst. Control Lett. **40**(2), 139–144 (2000)
42. Ruhe, A.: Rational Krylov sequence methods for eigenvalue computation. Linear Algebra Appl. **58**, 391–405 (1984)
43. Sabino, J.: Solution of large-scale Lyapunov equations via the block modified Smith method. ProQuest LLC, Ann Arbor, MI, (2007). Thesis (Ph.D.)–Rice University
44. Saff, E. B., Totik, V.: Logarithmic potentials with external fields, volume 316 of Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. (Berlin: Springer-Verlag, 1997). Appendix B by Thomas Bloom
45. Simoncini, V.: A new iterative method for solving large-scale Lyapunov matrix equations. SIAM J. Sci. Comput. **29**(3), 1268–1288 (2007)
46. Simoncini, V.: Computational methods for linear matrix equations. SIAM Rev. **58**(3), 377–441 (2016)
47. Starke, G.: Near-circularity for the rational Zolotarev problem in the complex plane. J. Approx. Theory **70**(1), 115–130 (1992)
48. Strang, G.: A proposal for Toeplitz matrix calculations. Stud. Appl. Math. **74**, 171–176 (1986)
49. Trefethen, L.N., Weideman, J.A.C.: The exponentially convergent trapezoidal rule. SIAM Rev. **56**(3), 385–458 (2014)
50. Wachspress, E.: The ADI Model Problem. Springer, New York (2013)
51. Wu, S.-L.: Toward parallel coarse grid correction for the parareal algorithm. SIAM J. Sci. Comput. **40**(3), A1446–A1472 (2018)
52. Zolotarev, E.: Application of elliptic functions to questions of functions deviating least and most from zero. Zap. Imp. Akad. Nauk. St. Petersburg **30**(5), 1–59 (1877)