BIT

Rank-adaptive dynamical low-rank integrators for first-order and second-order matrix differential equations

Marlis Hochbruck[1] · Markus Neher[1] · Stefan Schrammer[1]

Abstract

Dynamical low-rank integrators for matrix differential equations recently attracted a lot of attention and have proven to be very efficient in various applications. In this paper, we propose a novel strategy for choosing the rank of the projector-splitting integrator of Lubich and Oseledets adaptively. It is based on a combination of error estimators for the local time-discretization error and for the low-rank error with the aim to balance both. This ensures that the convergence of the underlying time integrator is preserved. The adaptive algorithm works for projector-splitting integrator methods for first-order matrix differential equations and also for dynamical low-rank integrators for second-order equations, which use the projector-splitting integrator method in its substeps. Numerical experiments illustrate the performance of the new integrators.

✉ Marlis Hochbruck
    marlis.hochbruck@kit.edu

    Markus Neher
    markus.neher@kit.edu

    Stefan Schrammer
    stefan.schrammer@kit.edu

1   Institute for Applied and Numerical Mathematics, Karlsruhe Institute of Technology, Karlsruhe, Germany

⚏ Springer

1 Introduction

Dynamical low-rank integrators [17, 19] have been designed for the approximation of large, time-dependent matrices which are solutions to first-order matrix differential equations

$$A'(t) = F\big(A(t)\big), \quad t \in [0, T], \quad A(0) = A_0 \in \mathbb{C}^{m \times n}, \tag{1}$$

whose solutions can be well approximated by low-rank matrices. The projector-splitting integrator introduced in [19] has particularly favorable properties. It is robust in the presence of small singular values, which appear in the case of over-approximation, i.e., when the approximation rank is chosen larger than the rank of the solution $A(t)$ of (1). A variant of this method adapted to strongly dissipative problems was presented in [5]. Another variant for stiff first-order matrix differential equations was introduced in [24].

Recently, a novel dynamical low-rank integrator for second-order matrix differential equations

$$A''(t) = F\big(A(t)\big), \quad t \in [0, T], \quad A(0) = A_0, \quad A'(0) = B_0, \tag{2}$$

has been constructed in [12]. Is is based on a Strang splitting of the equivalent first-order formulation and the projector-splitting integrator [19]. This integrator is also robust in the case of over-approximation and shows second-order convergence in time. In particular, with a few amendments it is an effective method for semilinear second-order matrix differential equations, see Sect. 2.3.

In applications, rank-adaptivity turns out to be essential for the efficiency of the algorithms. For first-order equations, in [4] a rank-adaptive variant of the unconventional integrator [5] was proposed. However, the approach from [4] is not applicable to the projector-splitting integrator [19]. In [8], rank-adaptivity for tensor methods for high-dimensional PDEs was based on a functional tensor train series expansion. For the special case of finite-dimensional parametrized Hamiltonian systems modeling non-dissipative phenomena, a rank-adaptive structure-preserving reduced basis method was introduced in [11]. Very recently, in [10] a Predictor–Corrector strategy for adaptivity was proposed, and in [31] the authors developed a rank-adaptive splitting method for the extended Fisher–Kolmogorov equation.

In the present paper, we discuss a general strategy for selecting the rank adaptively in the projector-splitting integrator. Increasing or decreasing the rank from one time step to the next was already proposed in [19] and quite recently in [13]. Our main contribution is a strategy for which the time step of the underlying splitting method, i.e., the Lie-Trotter splitting for first-order and the Strang splitting for second-order problems, is the only input parameter. We determine the rank such that the error of the dynamical low-rank approximation does not spoil the order of the underlying splitting method applied to the full matrix differential equation. This is achieved by propagating one additional singular value which is used for accepting or rejecting the current time step and for selecting the rank in the next step. The decision is based on an estimator of the global time-discretization error. This adaptivity control is also applicable to

the dynamical low-rank integrators for stiff problems. The new dynamical low-rank integrator for (2) uses the projector-splitting integrator in the substeps of the Strang splitting, which allows to control the rank adaptively also for second-order equations. Moreover, it can be readily combined with the integrator from [5].

The paper is organized as follows: In Sect. 2, we briefly recall the projector-splitting integrator introduced in [19] and the LRLF scheme from [12]. Additionally, we sketch variants of both methods for (stiff) semilinear first-order and second-order differential equations, respectively. Section 3 is devoted to rank-adaptivity. In Sect. 4, numerical experiments illustrate the performance of the new schemes.

Throughout this paper, m, n, and r are natural numbers, where w.l.o.g. $m \geq n \gg r$. If $n > m$, we consider the equivalent differential equation for the transpose. By $\mathcal{M}_r$ we denote the manifold of complex $m \times n$ matrices with rank r,

$$\mathcal{M}_r = \{\widehat{Y} \in \mathbb{C}^{m \times n} \mid \operatorname{rank}(\widehat{Y}) = r\}.$$

The Stiefel manifold of $m \times r$ unitary matrices is denoted by

$$\mathscr{V}_{m,r} = \{U \in \mathbb{C}^{m \times r} \mid U^H U = I_r\},$$

where I_r is the identity matrix of dimension r and U^H is the conjugate transpose of U.

The singular value decomposition of a matrix $Y \in \mathbb{C}^{m \times n}$ is given by

$$Y = U \Sigma V^H, \quad U \in \mathscr{V}_{m,m}, \quad V \in \mathscr{V}_{n,n}, \quad \Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n) \in \mathbb{C}^{m \times n},$$

where $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$ are its singular values. It is well known that for $r < n$, the rank-r best-approximation to Y w.r.t. the Frobenius norm is

$$\widehat{Y}^{\text{best}} = U \widetilde{\Sigma} V^H = \widehat{U} \widehat{\Sigma} \widehat{V}^H,$$

where $\widetilde{\Sigma} = \operatorname{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$ and

$$\widehat{U} = U[I_r \ 0] \in \mathscr{V}_{m,r}, \quad \widehat{V} = V[I_r \ 0] \in \mathscr{V}_{n,r}, \quad \widehat{\Sigma} = \operatorname{diag}(\sigma_1, \ldots, \sigma_r).$$

The Frobenius norm is denoted by $\| \cdot \|$, and the Frobenius inner product by $\langle \cdot, \cdot \rangle$. The symbol $\bullet$ denotes the entrywise or Hadmard product of matrices. For a given step size τ we use the notation $t_k = k\tau$ for any k with $2k \in \mathbb{N}_0$.

2 Dynamical low-rank integrators with fixed rank

In this section we give a review on various low-rank integrators for first and second-order matrix differential equations.

2.1 First-order differential equations

In the dynamical low-rank approximation of the solution to first-order matrix differential equations (1), the approximation $\widehat{A} \approx A$ is determined as solution of the projected differential equation

$$\widehat{A}'(t) = P\big(\widehat{A}(t)\big)F\big(\widehat{A}(t)\big), \quad \widehat{A}(0) = \widehat{A}_0. \tag{3}$$

Here, $P\big(\widehat{A}(t)\big)$ denotes the orthogonal projector onto the tangent space of the low-rank manifold $\mathscr{M}_r$ at $\widehat{A}(t) \in \mathscr{M}_r$, i.e.,

$$P\big(\widehat{A}(t)\big)Z = Z\widehat{V}(t)\widehat{V}(t)^H - \widehat{U}(t)\widehat{U}(t)^H Z\widehat{V}(t)\widehat{V}(t)^H + \widehat{U}(t)\widehat{U}(t)^H Z,$$

cf. [17, Lemma 4.1], where $\widehat{A}(t) \in \mathscr{M}_r$ is decomposed in a non-unique fashion resembling the singular value decomposition into

$$\widehat{A}(t) = \widehat{U}(t)\widehat{S}(t)\widehat{V}(t)^H, \quad \widehat{U}(t) \in \mathscr{V}_{m,r}, \quad \widehat{V}(t) \in \mathscr{V}_{n,r}, \quad \widehat{S}(t) \in \mathbb{C}^{r \times r} \text{ invertible.} \tag{4}$$

For the initial value $\widehat{A}_0$, typically the rank-r best-approximation to $A(0)$ computed by a truncated SVD is used.

The dynamical low-rank integrator developed in [19], also called the *projector-splitting integrator*, is constructed by performing a Lie-Trotter splitting on the right-hand side of (3) and solving the three subproblems on the low-rank manifold. This approach yields an efficient time-stepping algorithm for computing the desired low-rank approximations. One time-step of the projector-splitting integrator is given in Algorithm 1.

Algorithm 1 Projector-splitting integrator for low-rank approximations to the solution $A(t)$ of (1), single time step, cf. [19, Section 3.2]

1: **function** PSI($\widehat{U}, \widehat{S}, \widehat{V}, r, \Delta A$)
2: {**input:** factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-r approximation $\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H \approx A(t)$ with $\widehat{U} \in \mathscr{V}_{m,r}, \widehat{V} \in \mathscr{V}_{n,r}$,
3:    $\widehat{S} \in \mathbb{C}^{r \times r}$, functions for matrix-vector multiplication with ΔA and ΔA^H, where
4:    $\Delta A = \tau F(\widehat{A})$ }
5:
6:   $\widetilde{K} = \Delta A \widehat{V}$
7:   $\widehat{K} = \widehat{U}\widehat{S} + \widetilde{K}$
8:   compute QR-decomposition $\widehat{U}\widehat{S} = \widehat{K}$
9:   $\widehat{S} = \widehat{S} - \widehat{U}^H \widetilde{K}$
10:   $\widehat{L} = \widehat{V}\widehat{S}^H + \Delta A^H \widehat{U}$
11:   compute QR-decomposition $\widehat{V}\widehat{S}^H = \widehat{L}$
12:
13:   **return** $\widehat{U}, \widehat{S}, \widehat{V}, \widehat{L}$
14:   {**output:** factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-r approximation $\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H \approx A(t + \tau)$ and $\widehat{L} = \widehat{V}\widehat{S}^H$ (optional),
15:     with $\widehat{U} \in \mathscr{V}_{m,r}, \widehat{V} \in \mathscr{V}_{n,r}, \widehat{S} \in \mathbb{C}^{r \times r}$ }
16: **end function**

2.2 Second-order differential equations

For the second-order problem (2), a novel dynamical low-rank integrator named LRLF (low-rank leapfrog) scheme was presented in [12, Section 3]. Given $r_A, r_B \in \mathbb{N}$ and approximations $\widehat{A}_k \approx A(t_k)$ of rank r_A and $\widehat{B}_{k-\frac{1}{2}} \approx A'(t_{k-\frac{1}{2}})$ of rank r_B, it computes approximations $\widehat{A}_{k+1} \in \mathscr{M}_{r_A}$ and $\widehat{B}_{k+\frac{1}{2}} \in \mathscr{M}_{r_B}$ with $\widehat{A}_{k+1} \approx A(t_{k+1})$ and $\widehat{B}_{k+\frac{1}{2}} \approx A'(t_{k+\frac{1}{2}})$, respectively. This integrator is based on the first-order formulation of (2),

$$\begin{bmatrix} A \\ B \end{bmatrix}' = \begin{bmatrix} 0 \\ F(A) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix},$$

combined with a Strang splitting. The subproblems are first-order matrix differential equations, (5a, 5b)

$$A' = B, \quad B' = 0, \tag{5a}$$

$$B' = F(A), \quad A' = 0, \tag{5b}$$

which can be solved exactly. The low-rank matrices $\widehat{A}_{k+1}$ and $\widehat{B}_{k+\frac{1}{2}}$ are obtained by approximating the solutions of (5a) by application of Algorithm 1 to

$$\widetilde{B}'_{k-\frac{1}{2}}(\sigma) = F(\widehat{A}_k), \quad \widetilde{B}_{k-\frac{1}{2}}(0) = \widehat{B}_{k-\frac{1}{2}}, \quad \sigma \in [0, \tau],$$
$$\widetilde{A}'_k(\sigma) = \widetilde{B}_{k+\frac{1}{2}}, \qquad \widetilde{A}_k(0) = \widehat{A}_k, \qquad \sigma \in [0, \tau].$$

This leads to the dynamical low-rank integrator LRLF shown in Algorithm 2.

2.3 Semilinear problems

A fixed-rank dynamical low-rank integrator for the stiff semilinear first-order problem

$$A'(t) = L_1 A(t) + A(t)L_2 + f\big(A(t)\big), \quad t \in [0, T], \quad A(0) = A_0, \tag{6}$$

where the norms of $L_1 \in \mathbb{C}^{m \times m}$ and $L_2 \in \mathbb{C}^{n \times n}$ are large and f is a Lipschitz continuous function with moderate Lipschitz constant, was introduced in [24]. It is based on the subproblems (7a, 7b)

$$A' = L_1 A + A L_2, \tag{7a}$$

$$A' = f(A). \tag{7b}$$

The solution to (7a) is given by

$$A(t) = \exp(t L_1) A_0 \exp(t L_2). \tag{8}$$

Algorithm 2 DLR integrator for second-order ODEs (2), LRLF scheme, single time step, cf. [12, Algorithm 2]

1: **function** LRLF($\tau, F, \widehat{U}, \widehat{S}, \widehat{V}, \widehat{T}, \widehat{R}, \widehat{W}, r_A, r_B$)
2:    {**input:**   step size τ, right-hand side F,
3:            factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-r_A approximation $\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H \approx A(t)$ with $\widehat{U} \in \mathscr{V}_{m,r_A}$, $\widehat{V} \in$
   $\mathscr{V}_{n,r_A}$,
4:            $\widehat{S} \in \mathbb{C}^{r_A \times r_A}$,
5:            factors $\widehat{T}, \widehat{R}, \widehat{W}$ of rank-r_B approximation $\widehat{B} = \widehat{T}\widehat{R}\widehat{W}^H \approx A'(t - \frac{\tau}{2})$ with $\widehat{T} \in \mathscr{V}_{m,r_B}$,
6:            $\widehat{W} \in \mathscr{V}_{n,r_B}$, $\widehat{R} \in \mathbb{C}^{r_B \times r_B}$ }
7:
8:    $\widehat{B}$-step: $\widehat{T}, \widehat{R}, \widehat{W}, L = \mathrm{PSI}(\widehat{T}, \widehat{R}, \widehat{W}, r_B, \Delta B)$ where $\Delta B = \tau F(\widehat{U}\widehat{S}\widehat{V}^H)$
9:
10:    $\widehat{A}$-step: $\widehat{U}, \widehat{S}, \widehat{V} = \mathrm{PSI}(\widehat{U}, \widehat{S}, \widehat{V}, r_A, \Delta A)$ where $\Delta A = \tau \widehat{T}\widehat{L}^H$
11:
12:    **return** $\widehat{U}, \widehat{S}, \widehat{V}, \widehat{T}, \widehat{R}, \widehat{W}$
13:    {**output:** factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-r_A approximation $\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H \approx A(t + \tau)$ with $\widehat{U} \in \mathscr{V}_{m,r_A}$,
14:            $\widehat{V} \in \mathscr{V}_{n,r_A}$, $\widehat{S} \in \mathbb{C}^{r_A \times r_A}$,
15:            factors $\widehat{T}, \widehat{R}, \widehat{W}$ of rank-r_B approximation $\widehat{B} = \widehat{T}\widehat{R}\widehat{W}^H \approx A'(t + \frac{\tau}{2})$ with $\widehat{T} \in \mathscr{V}_{m,r_B}$,
16:            $\widehat{W} \in \mathscr{V}_{n,r_B}$, $\widehat{R} \in \mathbb{C}^{r_B \times r_B}$ }
17: **end function**

Note that the rank of the initial value is preserved for all times [24, Section 3.2]. In contrast, the rank of the solution to the nonlinear subproblem (7b) may vary in time.

In [24], a low-rank approximation has been computed by applying a Lie-Trotter splitting to (6) and solving the subproblems (7) by the projector-splitting integrator in Algorithm 1. This method is called PSI-stiff in the following.

For semilinear second-order equations of the form

$$A''(t) = -\Omega_1^2 A(t) - A(t)\Omega_2^2 + f(A), \quad t \in [0, T], \quad A(0) = A_0, \quad A'(0) = B_0, \tag{9}$$

with Hermitian, positive semidefinite matrices $\Omega_1 \in \mathbb{C}^{m \times m}$, $\Omega_2 \in \mathbb{C}^{n \times n}$ and f again Lipschitz continuous, a dynamical low-rank integrator named LRLF-semi was proposed in [12, Section 5]. It is based on the equivalent first-order formulation of the second-order problem (9), where the right-hand side is split into

$$\begin{bmatrix} A \\ B \end{bmatrix}' = \begin{bmatrix} B \\ -\Omega_1^2 A - A\Omega_2^2 + f(A) \end{bmatrix} = \begin{bmatrix} \omega_1^2 B \\ -\Omega_1^2 A \end{bmatrix} + \begin{bmatrix} \omega_2^2 B \\ -A\Omega_2^2 \end{bmatrix} + \begin{bmatrix} \omega_3^2 B \\ f(A) \end{bmatrix}. \tag{10}$$

The weights $\omega_i \geq 0$, $i = 1, 2, 3$, can be chosen arbitrarily such that $\omega_1^2 + \omega_2^2 + \omega_3^2 = 1$. A natural choice is $\omega_i^2 = 1/3$. The linear subproblems can be solved exactly. Low-rank approximations to these solutions are obtained by application of the projector-splitting integrator. The nonlinear subproblem is solved approximately with a variant of the LRLF scheme, cf. [12, Algorithm 3]. Denoting the numerical flows of the linear subproblems by $\phi_\tau^{\Omega_1}$ and $\phi_\tau^{\Omega_2}$, and the numerical flow of the nonlinear subproblem as $\phi_\tau^{\mathscr{I}}$, respectively, one step of the LRLF-semi scheme reads

$$\begin{bmatrix} \widehat{A}_{k+1} \\ \widehat{B}_{k+1} \end{bmatrix} = \left(\phi_{\frac{\tau}{2}}^{\Omega_1} \circ \phi_{\frac{\tau}{2}}^{\Omega_2} \circ \phi_{\tau}^{\mathscr{S}} \circ \phi_{\frac{\tau}{2}}^{\Omega_2} \circ \phi_{\frac{\tau}{2}}^{\Omega_1} \right) \begin{bmatrix} \widehat{A}_k \\ \widehat{B}_k \end{bmatrix}.$$

3 Rank adaptivity

In many applications, an appropriate rank for computing a low-rank approximation to the exact solution of (2) or (1) is not known a priori and it may also vary with time. If the rank is chosen too small, the low-rank approximation lacks accuracy. Conversely, if the rank is chosen too large, the algorithm becomes inefficient.

In the following, we develop rank-adaptive variants of the projector-splitting integrators PSI and PSI-stiff for first-order problems and for the LRLF and the LRLF-semi schemes for second-order problems.

3.1 Selecting the rank

We first discuss the projector-splitting integrator for the first-order problem (1). The general idea of our rank-adaptive strategy is to approximate the exact solution of (1) by a low-rank solution of rank r_k in the kth time step, but to propagate a solution of rank $r_k + 1$. The additional information is used as an indicator whether to accept or reject the current time step, and for selecting the rank r_{k+1} for the next time step.

Given $\widehat{A}_k = \widehat{U}_k \widehat{S}_k \widehat{V}_k^H \in \mathcal{M}_{r_k+1}$, a single step of Algorithm 1 yields the approximation $\widehat{A}_{k+1} = \widehat{U}_{k+1} \widehat{S}_{k+1} \widehat{V}_{k+1}^H \in \mathcal{M}_{r_k+1}$. We then compute the singular value decomposition of $\widehat{S}_{k+1}$,

$$\widehat{S}_{k+1} = P_{k+1} \Sigma_{k+1} Q_{k+1}^H, \quad \Sigma_{k+1} = \mathrm{diag}(\widehat{\sigma}_1, \dots, \widehat{\sigma}_{r_k}, \widehat{\sigma}_{r_k+1}),$$

with $P_{k+1}, Q_{k+1} \in \mathscr{V}_{r_k+1, r_k+1}$, and $\widehat{\sigma}_1 \geq \cdots \geq \widehat{\sigma}_{r_k+1} \geq 0$ so that

$$\widehat{A}_{k+1} = (\widehat{U}_{k+1} P_{k+1}) \Sigma_{k+1} (\widehat{V}_{k+1} Q_{k+1})^H$$

is the singular value decomposition of $\widehat{A}_{k+1}$. Given a tolerance `tol`, we determine r_k such that

$$\widehat{\sigma}_{r_k+1} < \texttt{tol} \leq \widehat{\sigma}_{r_k} \tag{11}$$

by distinguishing three cases:

1. *Augmentation case* If $\widehat{\sigma}_{r_k+1} \geq$ `tol`, the step is rejected and recalculated with rank $r_k + 2$. The ranks of the initial values $\widehat{U}_k$, $\widehat{S}_k$, $\widehat{V}_k$ of the current integration step are increased by adding a zero entry to $\widehat{S}_k$,

$$S^* = \begin{bmatrix} \widehat{S}_k & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{C}^{(r_k+2) \times (r_k+2)}.$$

This choice has been motivated by [19, Section 5.2]. The matrices $\widehat{U}_k$ and $\widehat{V}_k$ are augmented by unit vectors $u \in \mathbb{C}^m$ and $v \in \mathbb{C}^n$ such that

$$U^* = \begin{bmatrix} \widehat{U}_k & u \end{bmatrix} \in \mathcal{V}_{m,r_k+2}, \quad V^* = \begin{bmatrix} \widehat{V}_k & v \end{bmatrix} \in \mathcal{V}_{n,r_k+2}.$$

Numerical tests indicate that choosing u and v as random vectors and orthonormalizing them against $\widehat{U}_k$ and $\widehat{V}_k$ is reliable and robust, but other choices are also possible. Clearly, $U^* S^* (V^*)^H = \widehat{U}_k \widehat{S}_k \widehat{V}_k^H = \widehat{A}_k$, thus the initial value of the current integration step has not changed. However, the numerical approximation has been enabled to evolve to rank $r_k + 2$. The step is recomputed with the new initial values U^*, S^*, V^*, and it is again checked if the new smallest singular value is sufficiently small for accepting the step. This procedure is repeated until (11) is satisfied and the step is finally accepted, see Algorithm 3 for details.

2. *Reduction case* If $\widehat{\sigma}_{r_k} < \texttt{tol}$, this indicates that a sufficiently accurate approximation is available with a smaller rank. The step is accepted, but the rank for the next step is set to

$$r_{k+1} = \max \left\{ \operatorname{argmin}\{ j \mid \widehat{\sigma}_{j+1} < \texttt{tol} \}, r_k - 2 \right\},$$

i.e., the rank is reduced by either 1 or 2. Thus, the rank may only decay slowly. Sudden rank-drops are prohibited. For the initial values in the next time step we use

$$\widetilde{S} = \widetilde{I}^T \Sigma_{k+1} \widetilde{I}, \quad \widetilde{U} = (\widehat{U}_{k+1} P_{k+1}) \widetilde{I}, \quad \widetilde{V} = (\widehat{V}_{k+1} Q_{k+1}) \widetilde{I},$$

where $\widetilde{I} = \begin{bmatrix} I_{r_{k+1}+1} \\ 0 \end{bmatrix} \in \mathbb{C}^{(r_k+1) \times (r_{k+1}+1)}$. To prevent rank-oscillations, rank reduction is prohibited within the first 10 steps after an augmentation step.

3. *Persistent case* If $\widehat{\sigma}_{r_k} \geq \texttt{tol} > \widehat{\sigma}_{r_k+1}$, the time step is accepted and the same rank $r_{k+1} = r_k$ is used for the next one.

3.2 Choice of tolerance

It remains to find a suitable tolerance parameter $\texttt{tol}$. The global error of our integrator is a combination of a time-discretization error and a low-rank approximation error. We suggest to choose the rank such that the low-rank error is of about the same size as the time discretization error, but does not exceed the latter. If the time discretization error is large, the low-rank error is allowed to be large as well, and hence the approximation rank might be chosen small. If the time discretization error is small, then the approximation rank needs to be sufficiently large for an equally small low-rank error. By this procedure, we hope to ensure that the rank-adaptivity does not impair the convergence order of the integrator.

To balance the low-rank error with the time discretization error, we need to approximate the global time discretization error. This comprises an estimate of the local error

---

Algorithm 3 Augmentation

1: **function** AUGMENTATION($\widehat{U}, \widehat{S}, \widehat{V}, \Delta A, r, \text{tol}$)
2:     {**input:** factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-$(r+1)$ approximation $\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H$ with $\widehat{U} \in \mathcal{V}_{m,r+1}, \widehat{V} \in \mathcal{V}_{n,r+1}$,
3:         $\widehat{S} \in \mathbb{C}^{(r+1)\times(r+1)}$, functions for products with ΔA, tolerance tol}
4:
5:     ready = **False**
6:     **while not** ready **do**
7:         $r = r + 1$
8:         choose $u \in \mathbb{C}^m$ orthonormal to $\widehat{U}$ (e.g., random)
9:         choose $v \in \mathbb{C}^n$ orthonormal to $\widehat{V}$ (e.g., random)
10:       compute $\widehat{U} = U^*, S = S^*, V = V^*$ as in (12)
11:       $\widehat{U}, \widehat{S}, \widehat{V} = \text{PSI}(\widehat{U}, \widehat{S}, \widehat{V}, r+1, \Delta A)$
12:       compute singular values $\widehat{\sigma}_1, \ldots, \widehat{\sigma}_{r+1}$ of $\widehat{S}$
13:       ready = $(\widehat{\sigma}_{r+1} < \text{tol})$
14:     **end while**
15:     **return** $\widehat{U}, \widehat{S}, \widehat{V}, r$
16:     {**output:** factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-$(r+1)$ approximation to $\widehat{A} + \Delta A$ with $\widehat{U} \in \mathcal{V}_{m,r+1}, \widehat{V} \in \mathcal{V}_{n,r+1}$,
17:         $\widehat{S} \in \mathbb{C}^{(r+1)\times(r+1)}$}
18: **end function**

---

and a simple model for the global error. While the error analysis of the projector-splitting integrator [16] and of the LRLF scheme [12, Theorem 6] show exponential error growth w.r.t. time, numerical experiments indicate that this is a far too pessimistic bound, and that piecewise linear accretion in time is a more realistic scenario. Since the estimation of the local error induces some computational overhead, we keep an estimate for M steps before we recompute it. To be more precise, let e_ℓ be an approximation to the local error at time $t_{\ell M+1}$. Since we assume that the local error is constant for the next M time steps, the global error at time $t_{\ell M+j}$ is approximated by $E_\ell + je_\ell$, $j = 1, \ldots, M$, where E_ℓ is defined recursively by

$$E_{\ell+1} = E_\ell + Me_\ell, \quad \ell = 0, 1, \ldots, \quad E_0 = 0.$$

This simple technique worked very well in numerous numerical simulations. Note that the linear model is a conservative choice in the sense that if the error growth is faster than linear (e.g., quadratic or even exponential), then we underestimate the global error which enforces the integrator to use a larger rank. We thus still compute a numerical solution where the low-rank approximation does not impair the time integration error.

The tolerance threshold tol is then determined heuristically by the following steps:

1. Estimation of the local error (every M steps): Starting from an approximation $\widehat{A}_{\ell M} \approx A(t_{\ell M})$, we compute an approximation $\widehat{A}_{\ell M+1}$ to $A(t_{\ell M+1})$ by performing one integration step with step size τ and rank r. Additionally, we perform two time steps with step size $\frac{\tau}{2}$ and the same rank r, starting again from the initial value $\widehat{A}_{\ell M}$. By this, we obtain an alternative approximation $\breve{A}_{\ell M+1} \approx A(t_{\ell M+1})$. Assuming that the method converges with order $p \in \mathbb{N}$ in time, we apply Richardson extrapolation [9, Section II.4] to estimate the local error e_ℓ at $t_{\ell M+1}$ as

$$e_\ell := \frac{2^p}{2^p - 1} \|\widehat{A}_{\ell M+1} - \breve{A}_{\ell M+1}\|,$$

cf. [7, Section 5]. The global error is modeled as

$$\|A(t_{\ell M+j}) - \widehat{A}_{\ell M+j}\| \approx E_\ell + je_\ell, \quad j = 1, 2, \ldots, M. \tag{13}$$

2. Estimation of the low-rank error: If $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$ are the singular values of the exact solution $A(t_{k+1})$, then the rank-r_{k+1} best-approximation $\widehat{A}_{k+1}^{\text{best}}$ to $A(t_{k+1})$ fulfills

$$\frac{\|A(t_{k+1}) - \widehat{A}_{k+1}^{\text{best}}\|^2}{\|A(t_{k+1})\|^2} = \frac{\sigma_{r_{k+1}+1}^2 + \cdots + \sigma_n^2}{\sigma_1^2 + \cdots + \sigma_n^2} \leq \frac{(n - r_{k+1})\sigma_{r_{k+1}+1}^2}{\|\widehat{A}_{k+1}^{\text{best}}\|^2},$$

so that

$$\|A(t_{k+1}) - \widehat{A}_{k+1}^{\text{best}}\| \leq \sigma_{r_{k+1}+1} \frac{\|A(t_{k+1})\|}{\|\widehat{A}_{k+1}^{\text{best}}\|} \sqrt{n - r_{k+1}} \approx \widehat{\sigma}_{r_{k+1}+1} \sqrt{n - r_{k+1}}. \tag{14}$$

3. Tolerance threshold: The parameter $\texttt{tol}_k$ is set by equating the right-hand sides of (13) and (14) for $k = \ell M + j$, $j = 0, \ldots, M - 1$. This yields the condition

$$\widehat{\sigma}_{r_{k+1}+1} \leq \frac{E_\ell + je_\ell}{\sqrt{n - r_{k+1}}} =: \texttt{tol}_k, \quad k = \ell M + j. \tag{15}$$

4. Initial rank: An obvious choice for the initial rank for the integration is $r_0 = \text{rank}(A_0)$. However, if the rank of A_0 is very small, this may not necessarily hold for the rank of the exact solution $A(t)$, even for small t. On the other hand, if the rank of A_0 is large, this choice is also questionable. In our implementation, we first perform ν integration steps (with ν small, e.g., $\nu = 5$) with an initial rank r_1 given by the user (say $r_1 = 5$). Rank reduction is disabled in this phase. Then let r^* denote the number of singular values of $\widehat{A}_\nu$ greater than or equal to $\texttt{tol}_\nu$ defined in (15). If $r^* < r_1$, we continue the integration with $r_{\nu+1} = r^*$. Otherwise, we multiply r_1 by 2 and rerun the initializing process, until $r^* < r_1$ holds.

3.3 Rank-adaptive algorithms

The rank-adaptive version of the projector-splitting integrator Algorithm 1 is called RAPSI for *rank-adaptive projector-splitting integrator* in the following. A single step of the RAPSI scheme is given in Algorithm 4.

The rank-adaptive version of the LRLF scheme is derived by replacing the PSI routines by the RAPSI routines. We name this new integrator *rank-adaptive* LRLF (RALRLF) method. It is presented in Algorithm 5.

The rank-adaptive counterpart of the PSI-stiff scheme is named RAPSI-stiff. Since the linear subproblem preserves the rank, rank-adaptivity is only applied in the integration of the nonlinear subproblem (7b).

For semilinear second-order matrix differential equations of the form (9), we equip the integrator LRLF-semi with the adaptivity schemes described above. For the sake

Algorithm 4 Rank-adaptive projector-splitting integrator for first-order ODEs (1), single step

---

1: **function** RAPSI($\widehat{U}, \widehat{S}, \widehat{V}, r, \Delta A, \tau, p$)
2:    {**input:** factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-$(r+1)$ approximation $\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H \approx A(t)$ with $\widehat{U} \in \mathcal{V}_{m,r+1}$,
3:         $\widehat{V} \in \mathcal{V}_{n,r+1}$, $\widehat{S} \in \mathbb{C}^{(r+1)\times(r+1)}$, functions for products with ΔA, step size τ}
4:
5:    $\widehat{U}_1, \widehat{S}_1, \widehat{V}_1 = \text{PSI}(\widehat{U}, \widehat{S}, \widehat{V}, r+1, \Delta A)$
6:    compute SVD $\widehat{S}_1 = P\widehat{\sigma}Q^H$ where $\widehat{\sigma} = \text{diag}(\widehat{\sigma}_1, \ldots, \widehat{\sigma}_{r+1})$
7:    compute `tol` according to Sect. 3.2
8:    **if** $\widehat{\sigma}_r < $ `tol` **then**
9:       $r_1 = \text{argmin}\{j \mid \widehat{\sigma}_{j+1} < \text{tol}\}$
10:      $\widetilde{I} = [I_{r_1+1} \; 0]^T \in \mathbb{C}^{(r+1)\times(r_1+1)}$
11:      $\widehat{U}_1 = (\widehat{U}_1 P)\widetilde{I}$
12:      $\widehat{S}_1 = \widetilde{I}^T \widehat{\sigma} \widetilde{I}$
13:      $\widehat{V}_1 = (\widehat{V}_1 Q)\widetilde{I}$
14:    **else if** $\widehat{\sigma}_{r+1} \geq $ `tol` **then**
15:      $\widehat{U}_1, \widehat{S}_1, \widehat{V}_1, r_1 = \text{AUGMENTATION}(\widehat{U}, \widehat{S}, \widehat{V}, \Delta A, r, \text{tol})$
16:    **end if**
17:    **return** $\widehat{U}_1, \widehat{S}_1, \widehat{V}_1, r_1$, optional $L = \widehat{V}_1 \widehat{S}_1^H$
18:    {**output:** factors $\widehat{U}_1, \widehat{S}_1, \widehat{V}_1$ of rank-(r_1+1) approximation $\widehat{A}_1 = \widehat{U}_1 \widehat{S}_1 \widehat{V}_1^H \approx A(t+\tau)$ with
19:       $\widehat{U}_1 \in \mathcal{V}_{m,r_1+1}$, $\widehat{V}_1 \in \mathcal{V}_{n,r_1+1}$, $\widehat{S}_1 \in \mathbb{C}^{(r_1+1)\times(r_1+1)}$}
20: **end function**

---

of efficiency, rank changes are only implemented in the integration of the nonlinear subproblem, even though the linear subproblems are in general not rank-preserving. Only in the case of rank augmentation in the integration of the nonlinear subproblem, the affected substeps of [12, Algorithm 3] are recomputed. This adaptive integrator is named RALRLF-semi.

4 Numerical experiments

We now report on numerical experiments for matrix differential equations resulting from space discretizations of PDEs on a rectangular domain

$$\Omega = [-L_x, L_x] \times [-L_y, L_y] \subset \mathbb{R}^2. \tag{16}$$

For simplicity, we use a uniform mesh with n grid points in x- and m grid points in y-direction, i.e.,

$$\Omega_h = \{(x_j, y_i) \mid x_j = -L_x + jh_x, \; y_i = -L_y + ih_y, \; 0 \leq j \leq n, \; 0 \leq i \leq m\},$$
$$\text{with} \quad h_x = \frac{2L_x}{n}, \quad h_y = \frac{2L_y}{m}, \quad n, m \in \mathbb{N}. \tag{17}$$

Errors of the low-rank solutions are measured w.r.t. numerically computed reference solutions. Details are given in the respective subsections. Since we are only interested in the time discretization error, reference solutions and low-rank solutions are computed

Algorithm 5 Rank-adaptive integrator for second-order ODEs (2), RALRLF, full method

---

1: **function** RALRLF(τ, F, A_0, B_0)
2:     {**input:** step size τ, right-hand side F, initial values A_0, $B_0 \in \mathbb{C}^{m \times n}$}
3:
4:     compute initial ranks r_A, r_B according to Sect. 3.2
5:     compute rank-$(r_A + 1)$ best-approximation $\widehat{A} = \widehat{U} \widehat{S} \widehat{V}^H$, $\widehat{S} = \mathrm{diag}(\widehat{\sigma}_1, \ldots, \widehat{\sigma}_{r_A}, \widehat{\sigma}_{r_A+1})$ to A_0
6:     compute rank-$(r_B + 1)$ best-approximation $\widehat{B} = \widehat{T} \widehat{R} \widehat{W}^H$, $\widehat{R} = \mathrm{diag}(\rho_1, \ldots, \rho_{r_B}, \rho_{r_B+1})$ to B_0
7:     $t_0 = 0$
8:     **for** $k = 1, \ldots, n$ **do**
9:         $t_k = t_{k-1} + \tau$
10:
11:         $\widehat{B}$-*step:*
12:         $\widehat{T}, \widehat{R}, \widehat{W}, r_B, \widehat{L} = \mathrm{RAPSI}(\widehat{T}, \widehat{R}, \widehat{W}, r_B, \Delta B, \tau, 2)$ where $\Delta B = \begin{cases} \frac{\tau}{2} F(\widehat{U} \widehat{S} \widehat{V}^H), & k = 1, \\ \tau F(\widehat{U} \widehat{S} \widehat{V}^H), & \text{else} \end{cases}$
13:
14:         $\widehat{A}$-*step:*
15:         $\widehat{U}, \widehat{S}, \widehat{V}, r_A = \mathrm{RAPSI}(\widehat{U}, \widehat{S}, \widehat{V}, r_A, \Delta A, \tau, 2)$ where $\Delta A = \tau \widehat{T} \widehat{L}^H$
16:     **end for**
17:     $r = r_A$
18:     compute SVD $\widehat{S} = P \widehat{\sigma} Q^H$, $\widehat{U} = (UP)\widetilde{I}$, $\widehat{V} = (VQ)\widetilde{I}$, $\widehat{S} = \widetilde{I}^T \widehat{\sigma} \widetilde{I}$ where $\widetilde{I} = \begin{bmatrix} I_r & 0 \end{bmatrix}^T \in \mathbb{C}^{(r+1) \times r}$
19:
20:     **return** $\widehat{U}, \widehat{S}, \widehat{V}$
21:     {**output:** factors $\widehat{U}, \widehat{S}, \widehat{V}$ of rank-r approximation to exact solution $A(t_n)$ of (2) with $\widehat{U} \in \mathscr{V}_{m,r}$,
22:             $\widehat{V} \in \mathscr{V}_{n,r}$, $\widehat{S} \in \mathbb{C}^{r \times r}$}
23: **end function**

---

on the same spatial grid. The computation of the tolerance threshold as explained in Sect. 3.2 is performed with $M = 100$, i.e., every 100 steps we perform four additional steps with step size $\frac{\tau}{2}$, so that we increase the computational effort by 4%. Coosing a smaller value of M may sometimes be advantageous to reduce the global error.

All algorithms have been implemented in Python and were performed on a computer with an Intel(R) Core(TM) i7-7820X @ 3.60GHz CPU and 128 GB RAM storage. The codes are available from [25].

4.1 Nonlinear fractional Ginzburg–Landau equation

The fractional Ginzburg–Landau equation describes a variety of physical phenomena, cf. [22, 23, 28]. Here, we consider the problem in two space dimensions [29] and with homogeneous Dirichlet boundary conditions. Discretization in space by the second-order fractional centered difference method [3] yields the stiff semilinear matrix differential equation

$$A'(t) = -D_x A(t) - A(t) D_y - (\kappa + \mathrm{i}\xi) A(t) \bullet \overline{A(t)} \bullet A(t) + \gamma A(t), \quad A(0) = A_0. \tag{18}$$

Here, D_x and D_y are symmetric Toeplitz matrices [18] with first rows

$$\frac{\nu + i\eta}{h_x^\alpha}\left[g_1^\alpha, g_2^\alpha, \ldots, g_{n-1}^\alpha\right] \quad \text{and} \quad \frac{\nu + i\eta}{h_y^\beta}\left[g_1^\beta, g_2^\beta, \ldots, g_{m-1}^\beta\right],$$

respectively. Further, $i = \sqrt{-1}$, $\nu, \kappa > 0$, $\eta, \xi, \gamma \in \mathbb{R}$, $1 < \alpha, \beta < 2$ denote given parameters, and

$$g_k^\mu = \frac{(-1)^{k-1}\Gamma(1+\mu)}{\Gamma(\mu/2 - k + 2)\Gamma(\mu/2 + k + 2)}, \quad \mu \in \{\alpha, \beta\}, \quad k \in \mathbb{Z},$$

where $\Gamma(\cdot)$ denotes the Gamma-function.

In [29], (18) was solved with the linearized second-order backward differential scheme (LBDF2). A fixed-rank dynamical low-rank integrator for (18) was proposed in [32], based on considerations from [24]. We compute low-rank solutions of (18) with PSI-stiff and RAPSI-stiff. The solution to the linear subproblem of (18), which is of form (8), is computed with the Krylov subspace method proposed in [18].

An efficient implementation of products $F(\widehat{A})E$ of the right-hand side $F(A)$ in (18) with a skinny matrix E is crucial. For the linear part, this is achieved by computing the matrix products in

$$-D_x \widehat{U}\widehat{S}\widehat{V}^H E - \widehat{U}\widehat{S}\widehat{V}^H D_y E$$

successively from the right to the left. The implementation of the cubic nonlinear part of F is more involved, see "Appendix A.2".

In our first experiment, we use the same parameter values as in [32], namely $L_x = L_y = 10$, $m = n = 512$, $\nu = \eta = \kappa = \xi = \gamma = 1$, $T = 1$, $\alpha = 1.2$, $\beta = 1.9$, and the initial value

$$(A_0)_{ij} = 2\operatorname{sech}(x_j)\operatorname{sech}(y_i)\,e^{3i(x_j + y_i)}, \quad i, j = 1, \ldots, m.$$

The (full-rank) reference solution is computed with LBDF2 on the same spatial grid, with time step size $\tau = 10^{-4}$. Figure 1 shows the relative global errors

$$\operatorname{err} = \frac{\|A - \widehat{A}\|}{\|A\|}$$

between the reference solution A and the respective low-rank solutions $\widehat{A}$ at $t = T$ for different step sizes τ. Convergence order one is observed for the PSI-stiff scheme. For large step sizes, the approximations computed with the RAPSI-stiff method exhibit large errors. This is explained by the behavior of the singular values, cf. Fig. 1 (right picture). The time-discretization error is overestimated in this experiment, so that the tolerance threshold becomes so large that the second largest singular value is discarded. The induced low-rank error is then of magnitude 10^{-1}. If the parameter M is reduced to 10, this unfortunate rank reduction vanishes. However, reducing M increases the workload for the updates of `tol`, while $M = 100$ worked well in all other experiments and also in this experiment for smaller step sizes.
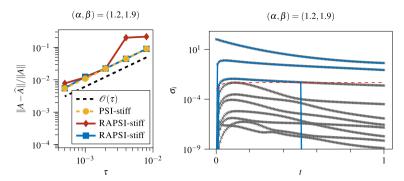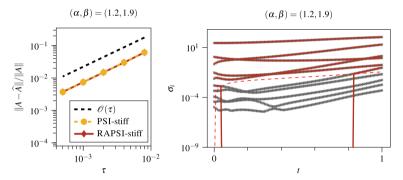
Fig. 1 Fractional Ginzburg–Landau equation, first experiment. The left picture shows the relative global error at $T = 1$ for $(\alpha, \beta) = (1.2, 1.9)$, where the fixed-rank approximation (yellow) is computed with $r = 5$. The rank-adaptive approximation was computed with $M = 100$ (orange) and $M = 10$ (blue). The trajectories of the ten largest singular values of the reference solution (gray), the singular values of RAPSI-stiff ($M = 10$) for $\tau = 1 \cdot 10^{-3}$ (blue), and the computed tolerance threshold (red, dashed) are displayed on the right (color figure online)

Fig. 2 Fractional Ginzburg–Landau equation, second experiment. The left picture shows the relative global error at $T = 1$ for $(\alpha, \beta) = (1.2, 1.9)$, where the fixed-rank approximation is computed with $r = 8$. The trajectories of the ten largest singular values of the reference solution (gray), the singular values of RAPSI-stiff for $\tau = 10^{-3}$ (orange), and the computed tolerance threshold (red, dashed) are displayed on the right (color figure online)

For our second example, we choose the second parameter set from [32], $L_x = L_y = 8$, $n = m = 512$, $\nu = \kappa = 1$, $\eta = 0.5$, $\xi = -5$, $\gamma = 3$, $\alpha = 1.2$, $\beta = 1.9$, and the initial values

$$(A_0)_{ij} = e^{-2(x_j^2 + y_i^2)} e^{i(S_0)_{ij}}, \quad \text{where} \quad (S_0)_{ij} = (e^{x_j + y_i} + e^{-(x_j + y_i)})^{-1},$$

for $i = 1, \ldots, m - 1$, $j = 1, \ldots, n - 1$. The relative global errors at $T = 1$ are displayed in Fig. 2. In contrast to the previous experiment, order one is observed for both integrators. Now the error curves for the fixed-rank integrator and its rank-adaptive variant align almost perfectly, and the singular values follow nicely the trajectories of the singular values of the reference solution.

Similar results for both experiments were obtained for the parameter values $(\alpha, \beta) = (1.5, 1.5)$, $(1.7, 1.3)$, and $(1.9, 1.2)$, cf. [25].
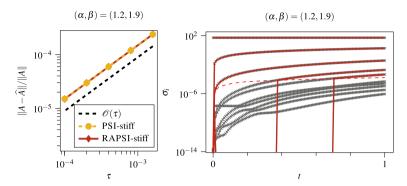
Fig. 3 Fractional Schrödinger equation. The left picture shows the relative global error at $T = 0.2$ for $(\alpha, \beta) = (1.2, 1.9)$, where the fixed-rank approximation is computed with $r = 5$. The trajectories of the ten largest singular values of the reference solution (gray), the singular values of RAPSI-stiff ($M = 100$) for $\tau = 4 \cdot 10^{-4}$ (orange), and the computed tolerance threshold (red, dashed) are displayed on the right (color figure online)

4.2 Nonlinear fractional Schrödinger equation

The nonlinear fractional Schrödinger equation [30] is a special case of the fractional Ginzburg–Landau equation (18) with $\nu = \kappa = \gamma = 0$. For the limit $\alpha, \beta \to 2$ it becomes the classical Schrödinger equation.

In our experiment, the reference solution to the problem was again computed with the LBDF2 method, using the step size $\tau = 2 \cdot 10^{-5}$. Figure 3 shows the results for the parameter values from [30], $L_x = L_y = 10$, $n = m = 512$, $\eta = 1$, $\xi = -2$, $T = 0.2$, $\alpha = 1.2$, $\beta = 1.9$, and the initial value

$$(A_0)_{ij} = \operatorname{sech}(x_j) \operatorname{sech}(y_i) \exp(\mathrm{i}(x_j + y_i)), \quad i, j = 1, \dots, m - 1.$$

Again, the relative global error curves match almost perfectly for both low-rank methods, and are also clearly indicating convergence of order one. The results for other choices of α and β are available in [25].

4.3 Laser-plasma interaction

As an example for second-order problems, we consider a reduced model of laser-plasma interaction from [14, 15, 26]. It is given by a wave equation with space-dependent cubic nonlinearity on a bounded, rectangular domain Ω given in (16) with periodic boundary conditions. After space discretization according to [26, Section 4.1.3], we obtain the matrix differential equation

$$A''(t) = \mathscr{L} A(t) - 0.3 \widetilde{\chi} \bullet \left(A(t) - \frac{1}{2} A(t) \bullet \overline{A(t)} \bullet A(t) \right) = F\big(A(t)\big), \quad (19)$$

with initial values $A(0) = A_0$ and $A'(0) = B_0$ given by

$$(A_0)_{ij} = 0.12 \exp\left(-\frac{y_i^2}{l_0^2} - \frac{x_j^2}{w_0^2} + \mathrm{i}y_i\right), \quad (B_0)_{ij} = \left(-\frac{2y_i}{l_0^2} - \mathrm{i}\right)(A_0)_{ij},$$

where x_j, y_i are defined in (17) and $i = 1, \ldots, m$, $j = 1, \ldots, n$. The discrete Laplacian $\mathscr{L}$ acts on $A(t)$ via

$$\mathscr{L}A(t) = \mathscr{F}_m^{-1}D_y^2\mathscr{F}_m A(t) + A(t)D_x,$$

where $D_x \in \mathbb{R}^{n \times n}$ denotes the symmetric Toeplitz matrix with first row

$$-\frac{1}{12h_x^2}[30, -16, 1, 0, \ldots, 0, 1, -16],$$

and

$$D_y = \frac{\mathrm{i}\pi}{L_y} \mathrm{diag}\left(0, \ldots, \frac{m}{2} - 1, -\frac{m}{2}, \ldots, -1\right).$$

$\mathscr{F}_m$ denotes the discrete Fourier transformation operator for m Fourier modes and $\mathscr{F}_m^{-1}$ its inverse. Hence we use fourth order finite differences with n equidistant grid points in transversal direction and a pseudospectral method with m equidistant grid points in longitudinal direction.

Equation (19) describes the propagation of a laser pulse with wavelength λ_0 in the direction of the positive y-axis through vacuum and through a strongly localized plasma barrier. The plasma is located between $y = 50\pi$ and $y = 300\pi$ and has constant density 0.3. The localization is modeled by the matrix $\widetilde{\chi} \in \mathbb{R}^{m \times n}$ with entries

$$\widetilde{\chi}_{ij} = \begin{cases} 1, & 50\pi \leq y_i \leq 300\pi, \\ 0, & \text{else.} \end{cases} \tag{20}$$

The interaction between the pulse and the plasma is modeled by a cubic nonlinearity. As in [15] we use the parameters $\lambda_0 = \pi$, $l_0 = 10\pi$, $w_0 = 100\pi$, $L_x = 300\pi$, and $L_y = 600\pi$.

Our numerical experiments were carried out for $t \in [0, 600\pi]$ with $n = 1024$ and $m = 8192$ discretization points in transversal and longitudinal direction, respectively. The reference solution was computed with the Gautschi-type method from [26], with step size $\tau_0 = L_y/(80m)$. For the low-rank solutions we used the step sizes $\tau = 2^k\tau_0$, $k = 2, \ldots, 6$. The algorithms LRLF and LRLF-semi were performed with fixed ranks $r_A = r_B = 4$.

In [12] it was oeberved that choosing the weights ω_i in LRLF-semi according to the direction of motion can improve the approximation significantly. Since the laser pulse moves mainly in longitudinal direction, we therefore used the weights

$$\omega_1^2 = \frac{2}{3}, \quad \omega_2^2 = 0, \quad \omega_3^2 = \frac{1}{3}$$
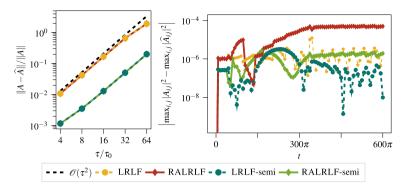
Fig. 4 Laser-plasma interaction. Relative global error between reference solution and low-rank approximations at $T = 600\pi$ (left), and absolute error in the maximal intensity for $\tau = 4\tau_0 = L_y/(20m)$ (right). The fixed-rank methods were computed with $r_A = r_B = 4$, the rank-adaptive methods with $M = 100$
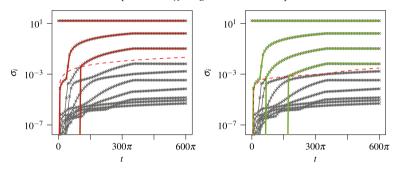

Fig. 5 Laser-plasma interaction. Trajectories of the ten largest singular values of the reference solution (gray) together with the trajectories of the singular values of the rank-adaptive low-rank integrators for $\tau = 4\tau_0$ (left: RALRLF, right: RALRLF-semi), and the respective computed tolerance thresholds (red, dashed) (color figure online)

in (10) for both LRLF-semi and RALRLF-semi.

The left picture in Fig. 4 shows the relative global error at $T = 600\pi$ between the reference solution and the different low-rank integrators. Second-order convergence is observed in all cases, and the integrators designed for semilinear problems yield better approximations than the other methods. The accuracy of the rank-adaptive schemes is comparable to those of the fixed-rank integrators, showing nicely that the heuristics works well for this example, cf. Fig. 5.

In physics, the maximal intensity $\max_{i,j} |A_{ij}(t)|^2$ of the propagating pulse over time is sometimes of higher interest than A itself. In the right picture of Fig. 4, the absolute error between the maximal intensity of the numerical pulse computed with the Gautschi-type integrator and the maximal intensity of the approximations obtained by the low-rank integrators is displayed.

4.4 Sine-Gordon equation

In our last experiment, we consider the two-dimensional sine-Gordon equation on the domain Ω from (16) with homogeneous Neumann boundary conditions [1]. Using

finite differences of second order on the grid (17) with $L_x = L_y = 7$, $n = m = 1001$ in both x- and y-direction, we obtain the semi-discretized matrix differential equation

$$A''(t) = DA(t) + A(t)D^T - \Phi \bullet \underline{\sin}(A(t)), \quad t \in [0, T], \quad A \in \mathbb{C}^{(m+1)\times(m+1)}.$$

Here, $\underline{\sin}(A)$ denotes the entrywise evaluation of the sine function. The matrix D is given by

$$D = \frac{1}{h^2} \begin{pmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 2 & -2 \end{pmatrix} \in \mathbb{R}^{(m+1)\times(m+1)}, \quad h = \frac{1}{m+1}.$$

The storage-economical evaluation of the products $\underline{\sin}(\widehat{A})E$ and $\underline{\sin}(\widehat{A})^H E$ is presented in "Appendix A". There is no preferred direction of propagation, so that we used the weights

$$\omega_1^2 = \omega_2^2 = \omega_3^2 = \frac{1}{3}$$

in (10) for the LRLF-semi and RALRLF-semi methods .

First we consider the initial values

$$(A_0)_{ij} = 4 \arctan \exp\left(\frac{x_j - 3.5}{0.954}\right), \quad (B_0)_{ij} = 0.629 \operatorname{sech}\left(\frac{x_j - 3.5}{0.954}\right),$$

and

$$\Phi_{ij} = 1 + \operatorname{sech}^2 \sqrt{x_j^2 + y_i^2},$$

$i, j = 0, \ldots, m$, of a line soliton in an inhomogeneous medium [2, Section 3.1.3]. The reference solution is computed by the leapfrog scheme on the same spatial grid with time step size $\tau = 2.5 \cdot 10^{-5}$. Figure 6 shows the relative global errors between the low-rank approximations and the reference solution. Convergence order two is observed for all methods. The fixed-rank integrators are slightly more accurate than their rank-adaptive pendants, probably because they use a higher rank.

In a second setting, we consider the symmetric perturbation of a static line soliton [2, Section 3.1.2] with $\Phi_{ij} = 1$, $(B_0)_{ij} = 0$, and

$$(A_0)_{ij} = 4 \arctan \exp\left(x_j + 1 - \frac{2}{\cosh(y_i + 7)} - \frac{2}{\cosh(y_i - 7)}\right), \quad i, j = 0, \ldots, m.$$

Figure 7 shows a similar behavior of the methods as in the first setting. For the RALRLF scheme however, the initial rank is rather large, and drops significantly after a few steps. This is caused in the routine for determining an appropriate initial rank. As explained
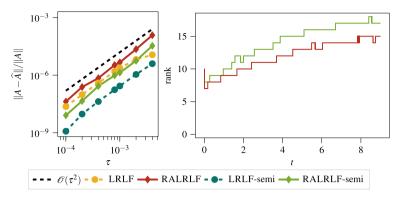
Fig. 6 Sine-Gordon equation, first setting. Relative global error at $T = 9$ between reference solution and low-rank approximations (left), and rank evolution of the solutions computed with the rank-adaptive integrators over time (right) for $\tau = 10^{-4}$. For the fixed-rank integrators, we used $r_A = r_B = 20$, for the rank-adaptive methods $M = 100$ (color figure online)
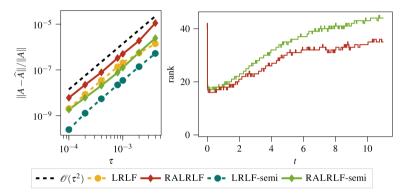

Fig. 7 Sine-Gordon equation, second setting. Relative global error at $T = 11$ between reference solution and low-rank approximations (left), and rank evolution of the solutions computed with the rank-adaptive integrators over time (right) for $\tau = 10^{-4}$. For the fixed-rank integrators, we used $r_A = r_B = 50$, for the rank-adaptive methods $M = 100$ (color figure online)

in Sect. 3.2, the initial guess $r_1 = 5$ is doubled repeatedly until the criterion for continuing the integration beyond the first ν steps is satisfied. In this experiment, an initial rank of ~ 23 is adequate. Therefore, the guesses 5, 10, and 20 are rejected, until $r_1 = 40$ is accepted and rank reduction applies in the subsequent integration steps.

5 Conclusion and outlook

In the present paper, we developed adaptive schemes for dynamical low-rank integrators for first and second-order matrix differential equations. The performance of these schemes have been illustrated by numerical experiments.

Both the projector-splitting integrator and the unconventional robust integrator have been successfully adapted to first-order tensor differential equations, cf. [6, 20, 21]

and references therein. We are confident that the strategy for the adaptive algorithm for matrix differential equations also works for the tensor case. This is part of ongoing research and will be reported in the future.

A Evaluation of entrywise functions for low-rank matrices

Let f be a (nonlinear) function that acts entrywise on matrices, let $E \in \mathbb{C}^{n \times r}$ be an arbitrary matrix, and let $\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H \in \mathcal{M}_r$ be a low-rank matrix with factors $\widehat{U}, \widehat{S}, \widehat{V}$ given in (4). For the efficiency of low-rank integrators for nonlinear problems, it is crucial to evaluate the product

$$f(\widehat{A})E, \tag{21}$$

using the factors $\widehat{U}, \widehat{S}$, and $\widehat{V}$ instead of the full matrix $\widehat{A}$. However, the entrywise computation of all components $\widehat{A}_{ij}$ of $\widehat{A}$ can not be avoided in general. Nevertheless, it is not necessary to store the full matrix $\widehat{A} \in \mathbb{C}^{m \times n}$, but it suffices to compute the rows of $\widehat{A}$ successively.

A.1 General functions

The ith row of $\widehat{A}$ is given by

$$(e_i \widehat{A}) = (e_i \widetilde{U})\widehat{V}^H \in \mathbb{C}^{1 \times n}, \quad i = 1, \ldots, m,$$

where $\widetilde{U} = \widehat{U}\widehat{S}$. The matrix products $(e_i \widetilde{U})\widehat{V}^H$ are carried out with complexity $\mathcal{O}(nr)$. We now evaluate f entrywise at $e_i \widehat{A}$ and multiply the result with E, which gives the ith row of the product (21). This sequence of operations can be performed for each $i = 1, \ldots, m$ independently and thus in parallel. This allows for a fast computation, even for large m. Similarly, the evaluation of

$$C = f(\widehat{A})^H E, \quad E \in \mathbb{C}^{m \times r},$$

is realized by computing the jth row via

$$(e_j C) = f(\widetilde{U}(e_j \widehat{V})^H)^H E, \quad j = 1, \ldots, n.$$

This approach is suitable for any function f acting entrywise on its argument, e.g., trigonometric functions. For monomials, the successive computation of the entries of $\widehat{A}$ can be avoided, which speeds up the computation further.

A.2 Monomials

Consider a (complex) monomial of degree $p \in \mathbb{N}$, $f(z) = z^p$ for $z \in \mathbb{C}$. The entrywise evaluation of f at $\widehat{A} \in \mathscr{M}_r$ is defined as $\big(f(\widehat{A})\big)_{ij} = \widehat{A}_{ij}^p$.

We now show how to compute the product (21) without computing or storing the elements of $\widehat{A}$ explicitly. Let $\widetilde{U} = \widehat{U}\widehat{S} \in \mathbb{C}^{m \times r}$. Then we have

$$\widehat{A} = \widehat{U}\widehat{S}\widehat{V}^H = \widetilde{U}\widehat{V}^H = \sum_{j=1}^{r} \widetilde{U}_j \widehat{V}_j^H, \tag{22}$$

where $\widetilde{U}_j$ is the jth column of $\widetilde{U}$ and $\widehat{V}_j$ the jth column of $\widehat{V}$, respectively. The Hadamard product is distributive and satisfies

$$(\widetilde{U}_j \widehat{V}_j^H) \bullet (\widetilde{U}_k \widehat{V}_k^H) = (\widetilde{U}_j \bullet \widetilde{U}_k)(\widehat{V}_j \bullet \widehat{V}_k)^H$$

for $1 \le j, k \le r$, cf. [27, Section 2]. Hence, for $E \in \mathbb{C}^{n \times r}$ it holds

$$\begin{aligned}
f(\widehat{A})E &= \left(\left(\sum_{j_1=1}^{r} \widetilde{U}_{j_1} \widehat{V}_{j_1}^H\right) \bullet \cdots \bullet \left(\sum_{j_p=1}^{r} \widetilde{U}_{j_p} \widehat{V}_{j_p}^H\right)\right) E \\
&= \left(\sum_{j_1,\ldots,j_p=1}^{r} (\widetilde{U}_{j_1} \bullet \widetilde{U}_{j_2} \bullet \cdots \bullet \widetilde{U}_{j_p})(\widehat{V}_{j_1} \bullet \widehat{V}_{j_2} \bullet \cdots \bullet \widehat{V}_{j_p})^H\right) E \\
&= \sum_{j_1,\ldots,j_p=1}^{r} (\widetilde{U}_{j_1} \bullet \widetilde{U}_{j_2} \bullet \cdots \bullet \widetilde{U}_{j_p})\big((\widehat{V}_{j_1} \bullet \widehat{V}_{j_2} \bullet \cdots \bullet \widehat{V}_{j_p})^H E\big). \tag{23}
\end{aligned}$$

For the special case of the cubic nonlinearity which appears in the examples in Sects. 4.1, 4.2, and 4.3,

$$f(\widehat{A}) = \widehat{A} \bullet \overline{\widehat{A}} \bullet \widehat{A},$$

where $\overline{\widehat{A}}$ denotes the complex conjugate of $\widehat{A}$, (23) reads

$$(\widehat{A} \bullet \overline{\widehat{A}} \bullet \widehat{A})E = \sum_{j,k,\ell=1}^{r} \widetilde{U}_{jk\ell} \widehat{V}_{jk\ell}^H E,$$

$$\text{where} \quad \widetilde{U}_{jk\ell} = \widetilde{U}_j \bullet \overline{\widetilde{U}}_k \bullet \widetilde{U}_\ell,$$
$$\widehat{V}_{jk\ell} = \widehat{V}_j \bullet \overline{\widehat{V}}_k \bullet \widehat{V}_\ell.$$

The computational cost is further reduced by exploiting the symmetry in j and ℓ,

$$(\widehat{A} \bullet \overline{\widehat{A}} \bullet \widehat{A})E = \sum_{k=1}^{r} \left[\sum_{j=1}^{r} (\widetilde{U}_j^2 \bullet \overline{\widetilde{U}}_k)\big((\widehat{V}_j^2 \bullet \overline{\widehat{V}}_k)^H E\big) + 2 \sum_{j=1}^{r} \sum_{\ell=1}^{j-1} \widetilde{U}_{jk\ell}(\widehat{V}_{jk\ell}^H E) \right]. \tag{24}$$

The product $(\widehat{A} \bullet \overline{\widehat{A}} \bullet \widehat{A})^H E$ with $E \in \mathbb{C}^{m \times r}$ can be computed analogously.

Additional simplifications apply to the product $(\widetilde{\chi} \bullet \widehat{A} \bullet \overline{\widehat{A}} \bullet \widehat{A})E$ with $\widetilde{\chi} \in \mathbb{C}^{m \times n}$ given in (20). It satisfies

$$\widetilde{\chi} = \begin{bmatrix} 0_{(\eta-1) \times n} \\ \mathbb{1}_{(\xi-\eta+1) \times n} \\ 0_{(m-\xi) \times n} \end{bmatrix} = \begin{pmatrix} 0_{\eta-1} \\ \mathbb{1}_{\xi-\eta+1} \\ 0_{m-\xi} \end{pmatrix} \mathbb{1}_n^T =: \widetilde{\mathbb{1}}_m \mathbb{1}_n^T, \tag{25}$$

where 0_n, $\mathbb{1}_n$ are the vectors of length n filled with zeros and ones, respectively, and $0_{n \times p}$ and $\mathbb{1}_{n \times p}$ the matrices of dimension $n \times p$ with all entries being zeros and ones, respectively. From (22) and (25), we obtain

$$(\widetilde{\chi} \bullet \widehat{A})E = \left[(\widetilde{\mathbb{1}}_m \mathbb{1}_n^T) \bullet \widehat{A} \right] E = \left[\sum_{j=1}^{r} \text{diag}(\widetilde{\mathbb{1}}_m) \widetilde{U}_j \widehat{V}_j^H \text{diag}(\mathbb{1}_n) \right] E$$

$$= \sum_{j=1}^{r} \begin{pmatrix} 0_{\eta-1} \\ \vartheta(\widetilde{U}_j) \\ 0_{m-\xi} \end{pmatrix} (\widehat{V}_j^H E) = \begin{bmatrix} 0_{(\eta-1) \times r} \\ \sum_{j=1}^{r} \vartheta(\widetilde{U}_j)(\widehat{V}_j^H)E \\ 0_{(m-\xi) \times r} \end{bmatrix},$$

where $\vartheta(\widetilde{U}_j)$ denotes the restriction of $\widetilde{U}_j$ to its ηth to ξth entries. Here, we made use of the following property of the Hadarmard product, cf. [27, Section 2]: If $A \in \mathbb{C}^{m \times n}$, $x \in \mathbb{R}^m$, and $y \in \mathbb{R}^n$, then

$$(x y^T) \bullet A = \text{diag}(x) A \, \text{diag}(y).$$

Hence, it suffices to compute and sum up the small matrices $\vartheta(\widetilde{U}_j)(\widehat{V}_j^H E)$. Likewise, using (24), we have

$$(\widetilde{\chi} \bullet \widehat{A} \bullet \overline{\widehat{A}} \bullet \widehat{A})E$$
$$= \begin{bmatrix} 0_{(\eta-1) \times r} \\ \sum_{k=1}^{r} \left[\sum_{j=1}^{r} (\vartheta(\widetilde{U}_j)^2 \bullet \vartheta(\overline{\widetilde{U}}_k))\big((\widehat{V}_j^2 \bullet \overline{\widehat{V}}_k)^H E\big) + 2 \sum_{j=1}^{r} \sum_{\ell=1}^{j-1} \vartheta(\widetilde{U}_{jk\ell})(\widehat{V}_{jk\ell}^H E) \right] \\ 0_{(m-\xi) \times r} \end{bmatrix}.$$

The implementation of $(\widetilde{\chi} \bullet \widehat{A} \bullet \overline{\widehat{A}} \bullet \widehat{A})^H E$, $E \in \mathbb{C}^{m \times r}$, is realized in the same manner.

References

1. Bratsos, A.G.: An explicit numerical scheme for the sine-Gordon equation in $2 + 1$ dimensions. Appl. Numer. Anal. Comput. Math. **2**(2), 189–211 (2005). https://doi.org/10.1002/anac.200410035
2. Bratsos, A.G.: The solution of the two-dimensional sine-Gordon equation using the method of lines. J. Comput. Appl. Math. **206**(1), 251–277 (2007). https://doi.org/10.1016/j.cam.2006.07.002
3. Çelik, C., Duman, M.: Crank-Nicolson method for the fractional diffusion equation with the Riesz fractional derivative. J. Comput. Phys. **231**(4), 1743–1750 (2012). https://doi.org/10.1016/j.jcp.2011.11.008
4. Ceruti, G., Kusch, J., Lubich, C.: A rank-adaptive robust integrator for dynamical low-rank approximation. BIT **62**, 1149–1174 (2022). https://doi.org/10.1007/s10543-021-00907-7
5. Ceruti, G., Lubich, C.: An unconventional robust integrator for dynamical low-rank approximation. BIT **62**, 23–44 (2022). https://doi.org/10.1007/s10543-021-00873-0
6. Ceruti, G., Lubich, C., Walach, H.: Time integration of tree tensor networks. SIAM J. Numer. Anal. **59**(1), 289–313 (2021). https://doi.org/10.1137/20M1321838
7. Constantinescu, E.M.: Generalizing global error estimation for ordinary differential equations by using coupled time-stepping methods. J. Comput. Appl. Math. **332**, 140–158 (2018). https://doi.org/10.1016/j.cam.2017.05.012
8. Dektor, A., Rodgers, A., Venturi, D.: Rank-adaptive tensor methods for high-dimensional nonlinear PDEs. J. Sci. Comput. **88**(2), 27 (2021). https://doi.org/10.1007/s10915-021-01539-3
9. Hairer, E., Nørsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I: Nonstiff Problems Springer. Series in Computational Mathematics, vol. 8, 2nd edn. Springer, Berlin (1993). https://doi.org/10.1007/978-3-540-78862-1
10. Hauck, C., Schnake, S.: A Predictor-Corrector Strategy for Adaptivity in Dynamical Low-Rank Approximations. https://arxiv.org/abs/2209.00550 (2022)
11. Hesthaven, J.S., Pagliantini, C., Ripamonti, N.: Rank-adaptive structure-preserving model order reduction of Hamiltonian systems. ESAIM Math. Model. Numer. Anal. **56**(2), 617–650 (2022). https://doi.org/10.1051/m2an/2022013
12. Hochbruck, M., Neher, M., Schrammer, S.: Dynamical low-rank integrators for second-order matrix differential equations. CRC 1173 Preprint 2022/12, Karlsruhe Institute of Technology. https://doi.org/10.5445/IR/1000143003. https://www.waves.kit.edu/downloads/CRC1173_Preprint_2022-12.pdf (2022)
13. Hu, J., Wang, Y.: An adaptive dynamical low rank method for the nonlinear Boltzmann equation. J. Sci. Comput. **92**(2), 24 (2022). https://doi.org/10.1007/s10915-022-01934-4
14. Karle, C., Schweitzer, J., Hochbruck, M., Laedke, E.W., Spatschek, K.H.: Numerical solution of nonlinear wave equations in stratified dispersive media. J. Comput. Phys. **216**(1), 138–152 (2006). https://doi.org/10.1016/j.jcp.2005.11.024
15. Karle, C., Schweitzer, J., Hochbruck, M., Spatschek, K.H.: A parallel implementation of a two-dimensional fluid laser-plasma integrator for stratified plasma-vacuum systems. J. Comput. Phys. **227**(16), 7701–7719 (2008). https://doi.org/10.1016/j.jcp.2008.04.024
16. Kieri, E., Lubich, C., Walach, H.: Discretized dynamical low-rank approximation in the presence of small singular values. SIAM J. Numer. Anal. **54**(2), 1020–1038 (2016). https://doi.org/10.1137/15M1026791
17. Koch, O., Lubich, C.: Dynamical low-rank approximation. SIAM J. Matrix Anal. Appl. **29**(2), 434–454 (2007). https://doi.org/10.1137/050639703
18. Lee, S.T., Pang, H.K., Sun, H.W.: Shift-invert Arnoldi approximation to the Toeplitz matrix exponential. SIAM J. Sci. Comput. **32**(2), 774–792 (2010). https://doi.org/10.1137/090758064
19. Lubich, C., Oseledets, I.V.: A projector-splitting integrator for dynamical low-rank approximation. BIT **54**(1), 171–188 (2014). https://doi.org/10.1007/s10543-013-0454-0
20. Lubich, C., Oseledets, I.V., Vandereycken, B.: Time integration of tensor trains. SIAM J. Numer. Anal. **53**(2), 917–941 (2015). https://doi.org/10.1137/140976546
21. Lubich, C., Vandereycken, B., Walach, H.: Time integration of rank-constrained Tucker tensors. SIAM J. Numer. Anal. **56**(3), 1273–1290 (2018). https://doi.org/10.1137/17M1146889

22. Milovanov, A., Rasmussen, J.J.: Fractional generalization of the Ginzburg–Landau equation: an unconventional approach to critical phenomena in complex media. Phys. Lett. A **337**(1), 75–80 (2005). https://doi.org/10.1016/j.physleta.2005.01.047

23. Mvogo, A., Tambue, A., Ben-Bolie, G.H., Kofané, T.C.: Localized numerical impulse solutions in diffuse neural networks modeled by the complex fractional Ginzburg-Landau equation. Commun. Nonlinear Sci. Numer. Simul. **39**, 396–410 (2016). https://doi.org/10.1016/j.cnsns.2016.03.008

24. Ostermann, A., Piazzola, C., Walach, H.: Convergence of a low-rank Lie–Trotter splitting for stiff matrix differential equations. SIAM J. Numer. Anal. **57**(4), 1947–1966 (2019). https://doi.org/10.1137/18M1177901

25. Schrammer, S.: Codes for numerical experiments (2022). https://doi.org/10.5445/IR/1000153539

26. Schweitzer, J.: Numerical Simulation of Relativistic Laser–Plasma Interaction. PhD thesis, Heinrich Heine University Düsseldorf. https://docserv.uni-duesseldorf.de/servlets/DocumentServlet?id=8401 (2008)

27. Styan, G.P.H.: Hadamard products and multivariate statistical analysis. Linear Algebra Appl. **6**, 217–240 (1973). https://doi.org/10.1016/0024-3795(73)90023-2

28. Tarasov, V.: Psi-series solution of fractional Ginzburg–Landau equation. J. Phys. A **39**, 8395–8407 (2006)

29. Zhang, Q., Lin, X., Pan, K., Ren, Y.: Linearized ADI schemes for two-dimensional space-fractional nonlinear Ginzburg–Landau equation. Comput. Math. Appl. **80**(5), 1201–1220 (2020). https://doi.org/10.1016/j.camwa.2020.05.027

30. Zhao, X., Sun, Z., Hao, Z.: A fourth-order compact ADI scheme for two-dimensional nonlinear space fractional Schrödinger equation. SIAM J. Sci. Comput. **36**(6), A2865–A2886 (2014). https://doi.org/10.1137/140961560

31. Zhao, Y.L., Gu, X.M.: An adaptive low-rank splitting approach for the extended Fisher–Kolmogorov equation. https://arxiv.org/abs/2210.15900 (2022)

32. Zhao, Y.L., Ostermann, A., Gu, X.M.: A low-rank Lie–Trotter splitting approach for nonlinear fractional complex Ginzburg–Landau equations. J. Comput. Phys. **446**, 12 (2021). https://doi.org/10.1016/j.jcp.2021.110652