Karlsruhe Institute of Technology

On dynamical low-rank integrators for matrix differential equations

Zur Erlangung des akademischen Grades eines

DOKTORS DER NATURWISSENSCHAFTEN

von der KIT-Fakultät für Mathematik des
Karlsruher Instituts für Technologie (KIT)
genehmigte

DISSERTATION

von

Stefan Schrammer

Tag der mündlichen Prüfung: 13. Juli 2022

1. Referentin: Prof. Dr. Marlis Hochbruck
2. Referent: PD Dr. Markus Neher
3. Referent: Assoz. Prof. Lukas Einkemmer, PhD

Acknowledgement

I gratefully acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 258734477 – SFB 1173.

I continue in German.

Mein erster Dank gebührt meiner Betreuerin Prof. Dr. Marlis Hochbruck. Nach meiner Rückkehr ans KIT im Anschluss an mein Referendariat hat sie Vertrauen in mich gesetzt und mich gefördert. Sie gab mir erste Einblicke in mathematische Forschung und nahm mich nach Ende meines Masterstudiums in ihre Gruppe auf. Ihre Art Mathematik zu vermitteln hat mich sehr geprägt, und für die Unterstützung damals wie heute bin ich ihr sehr dankbar.

Ebenso danken möchte ich PD Dr. Markus Neher, der überhaupt erst dazu geführt hat, dass ich mich für eine Rückkehr an die Universität entschieden habe. Er gab mir das Selbstvertrauen, das Ziel Promotion zu verfolgen. Seine zahlreichen Anmerkungen und Korrekturvorschläge haben meine Dissertation wesentlich verbessert, und ich bedanke mich bei ihm für seine Hilfe und seinen Rat während der letzten Jahre.

Dank gilt auch Assoz. Prof. Lukas Einkemmer, PhD, für die Bereitschaft die vorliegende Dissertation ebenfalls zu begutachten.

Sehr danken möchte ich ebenso PD Dr. Volker Grimm, der mich in einer schwierigen Phase mit seinem offenen Ohr und seiner Zeit entscheidend unterstützt hat. Der erfolgreiche Abschluss meiner Promotion ist auch sein Verdienst.

Zuletzt danke ich der gesamten erweiterten Arbeitsgruppe mit allen aktuellen und ehemaligen Kollegen für die gemeinsame Zeit und die moralische Unterstützung während meiner gesamten Promotion. Hervorzuheben sind dabei Benjamin und Jan, die sich nicht davor gescheut haben auch die vorläufigsten Versionen meiner Dissertation zu lesen.

Abstract

This thesis is concerned with dynamical low-rank integrators for matrix differential equations, typically stemming from space discretizations of partial differential equations. We first construct and analyze a dynamical low-rank integrator for second-order matrix differential equations, which is based on a Strang splitting and the projector-splitting integrator, a dynamical low-rank integrator for first-order matrix differential equations proposed by Lubich and Osedelets in 2014. For the analysis, we derive coupled recursive inequalities, where we express the global error of the scheme in terms of a time-discretization error and a low-rank error contribution. The first can be treated with Taylor series expansion of the exact solution. For the latter, we make use of an induction argument and the convergence result derived by Kieri, Lubich, and Walach in 2016 for the projector-splitting integrator.

From the original method, several variants are derived which are tailored to, e.g., stiff or highly oscillatory second-order problems. After discussing details on the implementation of dynamical low-rank schemes, we turn towards rank-adaptivity. For the projector-splitting integrator we derive both a technique to realize changes in the approximation ranks efficiently and a heuristic to choose the rank appropriately over time. The core idea is to determine the rank such that the error of the low-rank approximation does not spoil the time-discretization error. Based on the rank-adaptive pendant of the projector-splitting integrator, rank-adaptive dynamical low-rank integrators for (stiff and non-stiff) first-order and second-order matrix differential equations are derived. The thesis is concluded with numerical experiments to confirm our theoretical findings.

Contents

CHAPTER 1

Motivation and Introduction

Motivation

Many natural phenomena can be modeled by ordinary and partial differential equations. Their solutions help to give a precise description of a natural phenomenon. However, only for a limited number of problems the exact solution to the respective differential equation is known. Therefore, numerical schemes are used to compute approximations of sufficient accuracy to these exact solutions. For this, the problems, which are often continuous in time and space, need to be turned into finite dimensional problems. One approach is the method of lines, where first the spatial variable is discretized, while the time remains continuous. This yields a system of ordinary differential equations, which can be written as a matrix differential equation. In this thesis, we start from this situation, and only consider the integration in time.

The proper description of certain natural effects often requires a huge number of equations or a fine resolution in the discretizations in both space and time. Therefore, the matrix differential equations resulting from a discretization in space possess large dimensions. Although the computational power has increased significantly in the past, sometimes the sizes of the problems are too large for a standard integrator to yield approximations in a reasonable amount in time. Therefore, model reduction techniques have been derived in order to reduce the computational costs.

Dynamical low-rank integrators have been introduced in [Koch and Lubich, 2007] to approximate solutions of first-order matrix differential equations of type

$$A'(t) = F\big(A(t)\big), \qquad A(0) = A_0 \in \mathbb{C}^{m\times n},$$

which have large dimension but can be well-approximated by matrices of low rank. It was shown, that these matrices admit a factorization similarly to the singular value decomposition. Instead of working with matrices of full size $m \times n$, the new ansatz allows one to derive differential equations for the factors of the low-rank approximation, and working with the factors instead of the full matrices reduces both computational effort and required storage significantly. A drawback of the proposed method is its ill-conditioning in the presence of small singular values. This situation is commonly known as overapproximation and occurs if the approximation rank exceeds the true rank of the solution.

In [Lubich and Oseledets, 2014], a *projector-splitting integrator* was introduced, which is robust in the case of overapproximation. Together with a variant derived in [Ceruti and Lubich, 2021], it has been applied successfully to a variety of first-order matrix differential equations. Examples are the Vlasov–Poison equations in [Einkemmer and Lubich, 2018], Vlasov–Maxwell equations in [Einkemmer et al., 2020], Schrödinger equations in [Ceruti and Lubich, 2021], Burgers' equation with uncertainty in [Kusch et al., 2022], Boltzmann equations in [Kusch and Stammer, 2022], and many more. However, to the best of our knowledge, for second-order matrix differential equations

$$A''(t) = F\big(A(t)\big), \qquad A(0) = A_0, \quad A'(0) = B_0,$$

no dynamical low-rank integrator has been proposed so far.

Aims and main results

The main objectives of this thesis are the derivation, analysis, and implementation of dynamical low-rank integrators for computing low-rank approximations to the solutions of second-order matrix differential equations of the above form. The construction of these methods is based on the projector-splitting integrator derived in [Lubich and Oseledets, 2014]. The obvious strategy of reformulating the second-order differential equation into a first-order system and applying the projector-splitting integrator to compute approximations to A and A' behaved poorly in our numerical experiments. We thus propose to combine the projector-splitting integrator with a Strang splitting. The resulting scheme is named the St-LO method as an abbreviation for *Strang splitting combined with the integrator introduced by Lubich and Oseledets*. We shall see that this new method is closely related to the leapfrog scheme, one of the most popular numerical methods for computing approximations to second-order differential equations. Moreover, we perform a detailed error analysis of the St-LO scheme and show that the global error of the scheme is identical to the error of the leapfrog scheme, up to low-rank error contributions.

Based on the original method, we derive modifications of the St-LO scheme. This includes a variant which preserves certain geometric properties, and a method tailored to stiff second-order matrix differential equations. We also provide a dynamical low-rank integrator for highly oscillatory second-order differential equations which is derived from Gautschi-type schemes.

As another task, we consider rank-adaptivity for dynamical low-rank integrators for both first-order and second-order matrix differential equations. This has already been done in the past for the first-order case, e.g., [Ceruti et al., 2022; Dektor et al., 2021; Hesthaven et al., 2022]. However, the proposed ansatzes differ substantially and do not necessarily generalize to the dynamical low-rank integrators for second-order matrix differential equations. We therefore propose a new ansatz for choosing the rank adaptively. Based on an estimator of the time-discretization error, our strategy is designed such that the convergence in time is not impaired by the low-rank error. In other words, the rank-adaptive schemes are constructed in a way that they do not reduce the order of the underlying splitting method when applied to the full problem. This new ansatz is applicable to all dynamical low-rank integrators revised and constructed in this thesis.

Outline

This thesis is organized as follows. We first recall in Chapter 2 the most relevant topics of differential geometry, which are required for the construction of dynamical low-rank integrators for first-order matrix differential equations in Chapter 3. There, we also present variants of the original methods, which are either tailored to specific problems or posses unique properties.

Chapter 4 is devoted to the construction of dynamical low-rank integrators for second-order matrix differential equations based on the schemes discussed in Chapter 3. We construct the St-LO scheme and prove a second-order error bound in time. Moreover, we derive variants of the St-LO scheme for highly oscillatory and stiff problems, respectively.

In Chapter 5 we focus on details of the implementation of dynamical low-rank integrators. We especially illustrate how to implement the schemes such that no matrices of full dimension appear in any intermediate step.

We propose a novel ansatz for controlling the approximation rank of a dynamical low-rank integrator in Chapter 6. We explain in detail the heuristics which are the basis for the rank-adaptivity, and also provide information regarding the individual rank-adaptive dynamical low-rank integrators constructed from the new ansatz.

We finalize this thesis with numerical experiments for first-order and second-order matrix differential equations, cf. Chapter 7, to illustrate our theoretical findings as well as the performance of both fixed-rank and rank-adaptive dynamical low-rank integrators.

CHAPTER 2

Differential geometry for embedded submanifolds of the Euclidean space

In the derivation of dynamical low-rank integrators for first-order matrix differential equations, differential geometry plays an important role. We hence start this thesis with an overview of the most relevant topics utilized in the construction of the dynamical low-rank integrators for first-order and second-order problems, respectively. The solutions to the matrix differential equations which we consider in this thesis are in general complex matrices of size $m \times n$. Such matrices can be viewed as points or vectors in the real Euclidean space of dimension $2mn$. Imposing restrictions like symmetry, invertibility, or a fixed or constraint rank defines a subset of the Euclidean space. Often, these subsets are manifolds, commonly denoted by *embedded submanifolds*. We thus restrict ourselves to the embedded case and introduce further concepts like *tangent spaces* and *orthogonal projections* only for embedded submanifolds. Readers interested in a more general approach to differential geometry are referred to, e.g., [Lee, 2012]. Additional information on the interplay of numerics and differential geometry are presented, e.g., in [Absil et al., 2008].

Throughout, we mainly deal with two particular embedded submanifolds, namely the low-rank manifold $\mathcal{M}_r$ and the Stiefel manifold $\mathcal{V}_{m,r}$ introduced in Section 2.2. They are of high interest, since dynamical low-rank integrators search for approximations to the solutions of matrix differential equations in $\mathcal{M}_r$, whose elements can be partly characterized by elements in $\mathcal{V}_{m,r}$. Understanding both manifolds and their properties thus allows us to understand the construction of dynamical low-rank integrators for first-order matrix differential equations as it was done in [Koch and Lubich, 2007] and [Lubich and Oseledets, 2014], and also enables us to adapt the ideas used for first-order matrix differential equations to second-order matrix differential equations in Chapter 4.

The characterization of embedded submanifolds is possible in many ways, where some of them are more straight-forward than others. We follow the approach presented in [Boumal, 2020] and use so-called local defining functions to introduce submanifolds of the Euclidean space. For this, we first have to give a proper description of the Euclidean space. Afterwards, we introduce the manifolds $\mathcal{M}_r$ and $\mathcal{V}_{m,r}$. This is followed by the definition of tangent spaces, and we will construct the tangent spaces to both $\mathcal{M}_r$ and

$\mathcal{V}_{m,r}$. Lastly, we derive the orthogonal projector onto the tangent space to $\mathcal{M}_r$, as it plays a key role in the construction of all dynamical low-rank integrators considered in this thesis.

The following chapter is mainly based on [Boumal, 2020]. If not stated otherwise, the presented definitions and theorems are taken from there. However, in the literature mostly the real case is treated, while for us the complex case is more relevant. We hence transfer all results immediately to the complex case. This requires attention in the choice of subspaces of the Euclidean space and their dimension. The overall strategy remains the same.

2.1 Notation

Let $\mathcal{E}$ denote a *linear space* over the real numbers. Examples for $\mathcal{E}$ are $\mathbb{R}^n$, $\mathbb{R}^{m\times n}$ or $\mathrm{Sym}(n)$, which denote the space of real vectors of length n, real matrices of dimension $m \times n$, and real symmetric $n \times n$ matrices, respectively. Every finite dimensional linear space $\mathcal{E}$ is isomorphic to $\mathbb{R}^d$, where d is the dimension of $\mathcal{E}$. An *inner product* on a linear space $\mathcal{E}$ is denoted by $\langle \cdot, \cdot \rangle$, and the *norm* of some element $u \in \mathcal{E}$ is $\|u\| = \sqrt{\langle u, u\rangle}$. The standard inner product over $\mathbb{R}^n$ is given as

$$\langle u, v\rangle = u^T v = \sum_{j=1}^{n} u_j v_j$$

with associated norm

$$\|u\|_2 = \sqrt{\sum_{j=1}^{n} u_j^2}.$$

Likewise, the standard inner product for linear matrix spaces like $\mathbb{R}^{m\times n}$ is the so-called *Frobenius inner product* with the associated *Frobenius norm*,

$$\langle U, V\rangle_F = \mathrm{tr}(U^T V) = \sum_{i=1}^{m}\sum_{j=1}^{n} U_{ij} V_{ij}, \qquad \|U\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} U_{ij}^2}.$$

The notation $\mathrm{tr}(M) = \sum_i M_{ii}$ denotes the trace of a square matrix M, which is given as the sum of the diagonal entries. Throughout this thesis, we will mostly use the Frobenius norm for matrices and hence simply write $\|\cdot\|$ instead of $\|\cdot\|_F$. Whenever a different matrix norm is chosen we will indicate this separately.

Similarly, one can consider vectors and matrices over the complex field. We identify $\mathbb{C}^n$ with $\mathbb{R}^{2n}$ by separating the real and imaginary parts of a vector in $\mathbb{C}^n$ into two vectors is $\mathbb{R}^n$. Likewise, the set of complex matrices $\mathbb{C}^{m\times n}$ is a real linear space with dimension $2mn$. The standard inner product and norm are then given as

$$\langle U, V\rangle = \mathrm{Re}\,\mathrm{tr}(U^H V), \qquad \|U\| = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |U_{ij}|^2}.$$

Remark 2.1. Throughout, we treat all spaces $\mathcal{E}$ as real spaces. If the elements of $\mathcal{E}$ are complex valued, the dimension of the space is thus determined by counting the degrees of freedom for both the real and imaginary part of an element in $\mathcal{E}$. ◇

The *differential* of a function between (distinct) Euclidean spaces $\mathcal{E}$ and $\mathcal{E}'$ is the linear map

$$\mathrm{D}F(x) : \mathcal{E} \to \mathcal{E}', \quad \mathrm{D}F(x)[v] = \lim_{t\to 0} \frac{F(x+tv) - F(x)}{t} = \frac{\mathrm{d}}{\mathrm{d}t} F(x+tv)\Big|_{t=0}.$$

This allows one to introduce embedded submanifolds:

Definition 2.2. *Let $\mathcal{M}$ be a subset of a linear space $\mathcal{E}$ with dimension d. $\mathcal{M}$ is called* **(smooth) embedded submanifold** *of $\mathcal{E}$ if either one of the following holds:*

1. *$\mathcal{M}$ is an open subset of $\mathcal{E}$. Then we also call $\mathcal{M}$ an* **open submanifold**. *If $\mathcal{M} = \mathcal{E}$, we also call it a* **linear manifold**.

2. *For a fixed integer $k \geq 1$ and for each $x \in \mathcal{M}$ there exists a neighborhood U of x in $\mathcal{E}$ and a smooth function $h : U \to \mathbb{R}^k$ such that the following two conditions hold:*

   (a) If y is in U, then $h(y) = 0$ if and only if $y \in \mathcal{M}$, or equivalently

   $$\mathcal{M} \cap U = h^{-1}(0) = \{y \in U \mid h(y) = 0\},$$

   and

   (b) $\operatorname{rank} \mathrm{D}h(x) = k$.

   Such a function h is called a **local defining function** *for $\mathcal{M}$ at x.*

The linear space $\mathcal{E}$ is also called **embedding space**.

2.2 The manifolds $\mathcal{M}_r$ and $\mathcal{V}_{m,r}$

Next, we introduce two examples of embedded submanifolds. They will arise multiple times throughout this thesis.

Let

$$\mathcal{M}_r^{m,n} = \mathcal{M}_r = \left\{\mathbf{Y} \in \mathbb{C}^{m\times n} \mid \operatorname{rank} \mathbf{Y} = r < \min\{m,n\}\right\}.$$

In the literature, $\mathcal{M}_r$ is often called the set of fixed-rank matrices. We are mainly interested in the situation $r \ll \min\{m,n\}$, so we call $\mathcal{M}_r$ the *low-rank manifold* throughout.

Next, for $r \leq m$ we consider the set

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

This set is called the *Stiefel manifold.* For $r = m$ we have the special case of the group of unitary $m \times m$ matrices, while for $r = 1$ we have the unit sphere in $\mathbb{C}^m$. $\mathcal{V}_{m,r}$ is an embedded submanifold of $\mathbb{C}^{m\times r} \cong \mathbb{R}^{2mr}$.

The proofs that both $\mathcal{M}_r$ and $\mathcal{V}_{m,r}$ are indeed submanifolds can be found in [Boumal, 2020, Sections 7.5 and 7.3]. In both cases, the proof is done by construction of a local defining function, which by Definition 2.2 yields the stated result. Note that in [Boumal, 2020] only the real case is discussed. With the same strategy one can show that also in the complex case the sets $\mathcal{M}_r$ and $\mathcal{V}_{m,r}$ are embedded submanifolds. However, one has to be careful with the dimensions of the arising vector spaces.

2.3 Tangent spaces to embedded submanifolds

As we will encounter soon, not only embedded submanifolds will play an important role within this thesis, but also the so-called *tangent spaces* to such manifolds. They are especially of interest when one considers differential equations on embedded manifolds. In this section, we first give a proper definition of these spaces. Afterwards, we construct the tangent spaces to the manifolds $\mathcal{M}_r$ and $\mathcal{V}_{m,r}$.

Definition 2.3 ([Hairer et al., 2006, Section IV.5.1] and [Boumal, 2020, Definition 3.10])**.** *Let $\mathcal{M}$ be an embedded submanifold of the Euclidean space $\mathcal{E}$ with dimension k. For $x \in \mathcal{M}$ define*

$$\mathcal{T}_x\mathcal{M} = \left\{ v \in \mathbb{R}^k \;\middle|\; \begin{array}{l} \text{there exists a differentiable path } \gamma : (-\varepsilon, \varepsilon) \to \mathbb{R}^k \\ \text{with } \gamma(t) \in \mathcal{M} \text{ for all } t,\ \gamma(0) = x,\ \gamma'(0) = v \end{array} \right\}. \tag{2.1}$$

That is, v is in $\mathcal{T}_x\mathcal{M}$ if and only if there exists a smooth curve on $\mathcal{M}$ passing through x with velocity v.

We call $\mathcal{T}_x\mathcal{M}$ the **tangent space** *to $\mathcal{M}$ at x. Vectors in $\mathcal{T}_x\mathcal{M}$ are called* **tangent vectors** *to $\mathcal{M}$ at x. The dimension of $\mathcal{T}_x\mathcal{M}$ is called the* **dimension of $\mathcal{M}$** *and is denoted by* $\dim \mathcal{M}$.

With Definitions 2.2 and 2.3, one immediately gets the following result:

Corollary 2.4. *If $\mathcal{M}$ is an open submanifold of some Euclidean space $\mathcal{E}$, it holds that*

$$\dim \mathcal{M} = \dim \mathcal{T}_x\mathcal{M} = \dim \mathcal{E}.$$

Otherwise,

$$\dim \mathcal{M} = \dim \mathcal{T}_x\mathcal{M} = \dim \mathcal{E} - k,$$

where k is the rank of the differential $\mathrm{D}h(x)$ and h is any local defining function for $\mathcal{M}$.

Remark 2.5. For every embedded submanifold $\mathcal{M}$, the tangent space $\mathcal{T}_x\mathcal{M}$ is a linear subspace of the embedding Euclidean space $\mathcal{E}$. ◇

Example 2.6. The dimension of the low-rank manifold $\mathcal{M}_r$ is

$$\dim \mathcal{M}_r = 2r(m + n - r). \tag{2.2}$$

For the Stiefel manifold $\mathcal{V}_{m,r}$ we have

$$\dim \mathcal{V}_{m,r} = 2mr - r^2. \tag{2.3}$$

○

Remark 2.7. For fixed rank r, the dimension of $\mathcal{M}_r$ grows linearly in $(m + n)$, while the dimension of the embedding space $\mathbb{C}^{m\times n}$ grows as mn. As pointed out in [Boumal, 2020, Section 7.5], this means that large matrices of small rank can be encoded with a small amount of memory. For any $\mathbf{Y} \in \mathcal{M}_r$ one way is to choose the reduced singular value decomposition

$$\mathbf{Y} = \widetilde{\mathbf{U}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^H, \qquad \widetilde{\mathbf{U}} \in \mathcal{V}_{m,r}, \quad \widetilde{\mathbf{V}} \in \mathcal{V}_{n,r}, \quad \widetilde{\mathbf{\Sigma}} = \mathrm{diag}(\sigma_1, \dots, \sigma_r), \tag{2.4}$$

where $\sigma_1 \geq \cdots \geq \sigma_r > 0$ are the ordered singular values of $\mathbf{Y}$. Note that such a decomposition is unique up to complex signs, i.e., complex scalar factors of absolute value 1. For our purposes, it is more convenient to use decompositions like

$$\mathbf{Y} = \mathbf{U}\mathbf{S}\mathbf{V}^H, \qquad \mathbf{U} \in \mathcal{V}_{m,r}, \quad \mathbf{V} \in \mathcal{V}_{n,r}, \quad \mathbf{S} \in \mathbb{C}^{r\times r} \text{ invertible}. \tag{2.5}$$

The memory required to store the factorization in (2.5) is slightly larger compared to (2.4). Also, the factorization (2.5) is not unique. For arbitrary unitary matrices $Q, P \in \mathbb{C}^{r\times r}$

$$\mathbf{U}\mathbf{S}\mathbf{V}^H = \mathbf{U}QQ^H\mathbf{S}PP^H\mathbf{V}^H = (\mathbf{U}Q)(Q^H\mathbf{S}P)(\mathbf{V}P)^H = \widehat{\mathbf{U}}\widehat{\mathbf{S}}\widehat{\mathbf{V}}^H,$$

where $\widehat{\mathbf{U}} \in \mathcal{V}_{m,r}$ and $\widehat{\mathbf{V}} \in \mathcal{V}_{n,r}$ are elements of the respective Stiefel manifolds and $\widehat{\mathbf{S}}$ is invertible. The decomposition (2.5) is the basis for the construction of all numerical integrators in Chapter 3 and Chapter 4, respectively. ◇

The definition of a tangent space via smooth curves is in general not helpful to determine the tangent space to a given embedded submanifold $\mathcal{M}$ of $\mathcal{E}$. However, there is a way to construct tangent spaces based on local defining functions:

Theorem 2.8 ([Boumal, 2020, Theorem 3.8])**.** *Let $\mathcal{M}$ be an embedded submanifold of $\mathcal{E}$. Consider $x \in \mathcal{M}$ and the set $\mathcal{T}_x\mathcal{M}$ as given in (2.1). If $\mathcal{M}$ is an open submanifold, then $\mathcal{T}_x\mathcal{M} = \mathcal{E}$. Otherwise, $\mathcal{T}_x\mathcal{M} = \ker \mathrm{D}h(x)$, where h is any local defining function at x.*

For the two examples of embedded submanifolds $\mathcal{M}_r$ and $\mathcal{V}_{m,r}$ we now determine the respective tangent spaces. We start with the tangent space to $\mathcal{V}_{m,r}$.

Example 2.9. The tangent space to $\mathcal{V}_{m,r}$ is

$$\begin{aligned} \mathcal{T}_{\mathbf{X}}\mathcal{V}_{m,r} &= \{\mathbf{V} \in \mathbb{C}^{m\times r} \mid \mathbf{X}^H\mathbf{V} + \mathbf{V}^H\mathbf{X} = 0\} \\ &= \{\mathbf{V} \in \mathbb{C}^{m\times r} \mid \mathbf{X}^H\mathbf{V} \in \mathrm{Skew}(r)\}, \end{aligned} \tag{2.6}$$

where $\mathrm{Skew}(r)$ denotes the set of skew-Hermitian matrices of size r,

$$\mathrm{Skew}(r) = \{Z \in \mathbb{C}^{r\times r} \mid Z^H = -Z\}.$$

As pointed out in [Boumal, 2020, Section 7.3], sometimes it is convenient to parameterize tangent vectors of $\mathcal{V}_{m,r}$ explicitly. This explicit representation is derived by completing $\mathbf{X}$ to a unitary basis of $\mathbb{C}^{m\times m}$. However, the implicit characterization via (2.6) is sufficient for now.

The dimension of $\mathcal{T}_{\mathbf{X}}\mathcal{V}_{m,r}$ is

$$\dim \mathcal{T}_{\mathbf{X}}\mathcal{V}_{m,r} = \dim \mathcal{V}_{m,r} = 2mr - r^2, \tag{2.7}$$

which is a direct consequence of Corollary 2.4 together with (2.3). ○

Remark 2.10. Any element in $\mathrm{Skew}(r)$ has one single degree of freedom per diagonal element, since the diagonal is purely imaginary. The entries above the diagonal have two degrees of freedom and determine implicitly the entries below the diagonal. Hence, the dimension of $\mathrm{Skew}(r)$ is

$$\dim \mathrm{Skew}(r) = r + 2 \cdot \frac{r(r+1)}{2} = r^2. \tag{2.8}$$

◇

We introduce the short-hand notation

$$A'(t) = \frac{\mathrm{d}}{\mathrm{d}t}A(t)$$

for the derivative with respect to the time t of some matrix valued function $t \mapsto A(t)$. If $\mathbf{U}(t)$ denotes a time-dependent element of the Stiefel manifold $\mathcal{V}_{m,r}$, the time derivative $\mathbf{U}'(t)$ is contained in the tangent space to $\mathcal{V}_{m,r}$ at $\mathbf{U}(t)$:

Corollary 2.11. *Let* $\mathbf{U} : [0,T] \to \mathcal{V}_{m,r},\ t \mapsto \mathbf{U}(t)$ *be a smooth mapping. Then*

$$\mathbf{U}'(t) \in \mathcal{T}_{\mathbf{U}(t)}\mathcal{V}_{m,r}.$$

Proof. Since $\mathbf{U}$ is smooth, it holds by the product rule that

$$\mathbf{U}(t)^H\mathbf{U}(t) = I_r \qquad \Longrightarrow \qquad \frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{U}(t)^H\mathbf{U}(t)) = \mathbf{U}'(t)^H\mathbf{U}(t) + \mathbf{U}(t)^H\mathbf{U}'(t) = 0$$

for all t. Thus, $\mathbf{U}(t)^H\mathbf{U}'(t) \in \mathrm{Skew}(r)$, and the assertion follows from (2.6). □

Before we derive the tangent space to $\mathcal{M}_r$, we first want to give an impression of how tangent elements to some $\mathbf{Y} = \mathbf{USV}^H \in \mathcal{M}_r$ satisfying (2.5) may look like. For this, consider the smooth mappings

$$\mathbf{U} : \mathbb{R} \to \mathcal{V}_{m,r},\ t \mapsto \mathbf{U}(t), \qquad \mathbf{S} : \mathbb{R} \to \mathbb{C}^{r\times r},\ t \mapsto \mathbf{S}(t), \qquad \mathbf{V} : \mathbb{R} \to \mathcal{V}_{n,r},\ t \mapsto \mathbf{V}(t)$$

such that $\mathbf{U}(0) = \mathbf{U}$, $\mathbf{S}(0) = \mathbf{S}$, and $\mathbf{V}(0) = \mathbf{V}$. Then

$$\mathbf{Y}(t) = \mathbf{U}(t)\mathbf{S}(t)\mathbf{V}(t)^H \tag{2.9}$$

is smooth and $\mathbf{Y}(0) = \mathbf{Y}$. By the product rule, the velocity of $\mathbf{Y}(t)$ at zero is

$$\mathbf{Y}'(0) = \mathbf{U}'(0)\mathbf{S}(0)\mathbf{V}(0)^H + \mathbf{U}(0)\mathbf{S}'(0)\mathbf{V}(0)^H + \mathbf{U}(0)\mathbf{S}(0)\mathbf{V}'(0)^H,$$

and is hence contained in the tangent space $\mathcal{T}_{\mathbf{Y}(0)}\mathbf{M}$ by Definition 2.3. Due to Corollary 2.11, it holds that $\mathbf{U}'(0) \in \mathcal{T}_{\mathbf{U}(0)}\mathcal{V}_{m,r}$ and $\mathbf{V}'(0) \in \mathcal{T}_{\mathbf{V}(0)}\mathcal{V}_{n,r}$, respectively. Any $\mathbf{Z}$ satisfying

$$\mathbf{Z} = \delta U\mathbf{SV}^H + \mathbf{U}\delta S\mathbf{V}^H + \mathbf{US}\delta V^H, \qquad \delta U \in \mathcal{T}_{\mathbf{U}}\mathcal{V}_{m,r}, \quad \delta V \in \mathcal{T}_{\mathbf{V}}\mathcal{V}_{n,r}, \quad \delta S \in \mathbb{C}^{r\times r} \tag{2.10}$$

is therefore contained in the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ to $\mathcal{M}_r$ at $\mathbf{Y}$.

In the following, we show that in fact all elements in $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ satisfy (2.10). This is achieved by studying the extended tangent map introduced in [Koch and Lubich, 2007] for the real case. We adopt their ansatz but adapt it to the complex case. Again, this requires attention when determining the dimensions of the arising spaces.

Example 2.12 (*Tangent space to* $\mathcal{M}_r$)**.** For $\mathbf{Y} = \mathbf{USV}^H \in \mathcal{M}_r$ consider the extended tangent map of $(\mathbf{S},\mathbf{U},\mathbf{V}) \mapsto \mathbf{Y}$ as in (2.5),

$$\pi : \begin{cases} \mathbb{C}^{r\times r} \times \mathcal{T}_{\mathbf{U}}\mathcal{V}_{m,r} \times \mathcal{T}_{\mathbf{V}}\mathcal{V}_{n,r} \to \mathcal{T}_{\mathbf{Y}}\mathcal{M}_r \times \mathrm{Skew}(r) \times \mathrm{Skew}(r), \\ (\delta S, \delta U, \delta V) \qquad\qquad\quad \mapsto (\delta U\mathbf{SV}^H + \mathbf{U}\delta S\mathbf{V}^H + \mathbf{US}\delta V^H, \mathbf{U}^H\delta U, \mathbf{V}^H\delta V). \end{cases}$$

Obviously, this map is linear. Now assume $(\delta S, \delta U, \delta V) \in \ker \pi$, i.e.,

$$\begin{aligned} \delta U\mathbf{SV}^H + \mathbf{U}\delta S\mathbf{V}^H + \mathbf{US}\delta V^H &= 0, \\ \mathbf{U}^H\delta U &= 0, \\ \mathbf{V}^H\delta V &= 0. \end{aligned}$$

Multiplication of the first equation with $\mathbf{U}^H$ from the left and $\mathbf{V}$ from the right yields

$$0 = \mathbf{U}^H\delta U\mathbf{SV}^H\mathbf{V} + \mathbf{U}^H\mathbf{U}\delta S\mathbf{V}^H\mathbf{V} + \mathbf{U}^H\mathbf{US}\delta V^H\mathbf{V} = \mathbf{U}^H\mathbf{U}\delta S\mathbf{V}^H\mathbf{V} = \delta S$$

due to $\mathbf{U} \in \mathcal{V}_{m,r}$, $\mathbf{V} \in \mathcal{V}_{n,r}$, and the remaining two equations. Likewise, multiplication of the first equation with $\mathbf{V}\mathbf{S}^{-1}$ from the right gives $\delta U = 0$, while multiplication with $\mathbf{S}^{-1}\mathbf{U}^H$ from the left yields $\delta V = 0$. As a consequence, the kernel of π is trivial.

We now compare the dimensions of the vector spaces on both sides of the map π. From (2.7) we obtain

$$\dim(\mathbb{C}^{r\times r} \times \mathcal{T}_{\mathbf{U}}\mathcal{V}_{m,r} \times \mathcal{T}_{\mathbf{V}}\mathcal{V}_{n,r}) = 2r^2 + (2mr - r^2) + (2nr - r^2) = 2r(m+n). \tag{2.11}$$

Though we have no explicit representation of $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ yet, we conclude from Corollary 2.4 and (2.2) that

$$\dim \mathcal{T}_{\mathbf{Y}}\mathcal{M}_r = 2r(m+n-r),$$

which together with (2.8) yields

$$\dim(\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r \times \mathrm{Skew}(r) \times \mathrm{Skew}(r)) = 2r(m+n-r) + r^2 + r^2 = 2r(m+n). \tag{2.12}$$

Comparing (2.11) with (2.12) reveals that the dimensions of the spaces on both sides of π coincide. Since in addition the kernel is trivial, π is an isomorphism. Hence, all elements in the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ satisfy (2.10), and the tangent space to $\mathcal{M}_r$ at $\mathbf{Y} = \mathbf{U}\mathbf{S}\mathbf{V}^H$ reads

$$\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r = \{\delta U\mathbf{S}\mathbf{V}^H + \mathbf{U}\delta S\mathbf{V}^H + \mathbf{U}\mathbf{S}\delta V^H \mid \delta S \in \mathbb{C}^{r\times r},\ \delta U \in \mathcal{T}_{\mathbf{U}}\mathcal{V}_{m,r},\ \delta V \in \mathcal{T}_{\mathbf{V}}\mathcal{V}_{n,r}\}. \tag{2.13}$$

An alternative representation of $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ is derived in [Boumal, 2020, Section 7.5] based on the smooth curve (2.9) on $\mathcal{M}_r$, namely

$$\begin{aligned}\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r = \{\mathbf{U}\mathbf{M}\mathbf{V}^H + \widetilde{\mathbf{U}}\mathbf{V}^H + \mathbf{U}\widetilde{\mathbf{V}}^H \mid \mathbf{M} \in \mathbb{C}^{r\times r},\ \widetilde{\mathbf{U}} \in \mathbb{C}^{m\times r},\ \widetilde{\mathbf{V}} \in \mathbb{C}^{n\times r},\\ \text{and } \mathbf{U}^H\widetilde{\mathbf{U}} = 0 = \mathbf{V}^H\widetilde{\mathbf{V}}\}.\end{aligned} \tag{2.14}$$

The reformulation

$$\mathbf{U}\mathbf{M}\mathbf{V}^H + \widetilde{\mathbf{U}}\mathbf{V}^H + \mathbf{U}\widetilde{\mathbf{V}}^H = \begin{bmatrix}\mathbf{U} & \widetilde{\mathbf{U}}\end{bmatrix}\begin{bmatrix}\mathbf{M} & I_r\\ I_r & 0\end{bmatrix}\begin{bmatrix}\mathbf{V} & \widetilde{\mathbf{V}}\end{bmatrix}^H \tag{2.15}$$

shows that any element in $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ has rank at most $2r$.

Section 7.5 in [Boumal, 2020] also contains a third representation of $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$,

$$\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r = \left\{\begin{bmatrix}\mathbf{U} & \mathbf{U}_\perp\end{bmatrix}\begin{bmatrix}A & B\\ C & 0\end{bmatrix}\begin{bmatrix}\mathbf{V} & \mathbf{V}_\perp\end{bmatrix} \mid A, B, C \text{ are arbitrary}\right\}, \tag{2.16}$$

where $\begin{bmatrix}\mathbf{U} & \mathbf{U}_\perp\end{bmatrix}$ and $\begin{bmatrix}\mathbf{V} & \mathbf{V}_\perp\end{bmatrix}$ form unitary bases of $\mathbb{C}^{m\times m}$ and $\mathbb{C}^{n\times n}$, respectively. This representation is not used in practice, as the memory required to store $\mathbf{U}_\perp$ and $\mathbf{V}_\perp$ by far exceeds the memory required to store $\mathbf{U}$ and $\mathbf{V}$ if r is small compared to m, n. However, (2.16) sometimes comes in handy when one is interested in particular analytical calculations. ◦

Similarly to Corollary 2.11, we show that the time-derivative of an element $\mathbf{A}(t) \in \mathcal{M}_r$ is contained in the tangent space to $\mathcal{M}_r$ at $\mathbf{A}(t)$:

Corollary 2.13. *Let* $\mathbf{A} : [0,T] \to \mathcal{M}_r,\ t \mapsto \mathbf{A}(t)$ *be smooth. Then it holds*

$$\mathbf{A}'(t) \in \mathcal{T}_{\mathbf{A}(t)}\mathcal{M}_r.$$

Proof. For $\mathbf{A}(t) \in \mathcal{M}_r$ we have by (2.5) the decomposition

$$\mathbf{A}(t) = \mathbf{U}(t)\mathbf{S}(t)\mathbf{V}(t)^H,$$

where $\mathbf{U}(t) \in \mathcal{V}_{m,r}$, $\mathbf{V}(t) \in \mathcal{V}_{n,r}$, and $\mathbf{S}(t) \in \mathbb{C}^{r\times r}$ are smooth. By the product rule, we have

$$\begin{aligned}\mathbf{A}'(t) &= \frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{U}(t)\mathbf{S}(t)\mathbf{V}(t)^H) \\ &= \mathbf{U}'(t)\mathbf{S}(t)\mathbf{V}(t)^H + \mathbf{U}(t)\mathbf{S}'(t)\mathbf{V}(t)^H + \mathbf{U}(t)\mathbf{S}(t)\mathbf{V}'(t)^H. \end{aligned} \tag{2.17}$$

The derivatives of $\mathbf{U}$ and $\mathbf{V}$ satisfy $\mathbf{U}'(t) \in \mathcal{T}_{\mathbf{U}(t)}\mathcal{V}_{m,r}$ and $\mathbf{V}'(t) \in \mathcal{T}_{\mathbf{V}(t)}\mathcal{V}_{n,r}$ due to Corollary 2.11, while $\mathbf{S}'(t) \in \mathbb{C}^{r\times r}$ is a square matrix. Hence, (2.17) satisfies the representation (2.13) of elements in the tangent space to $\mathcal{M}_r$, which proves the claim. □

2.4 Orthogonal projectors onto tangent spaces

Our last topic within the field of differential geometry is the construction of orthogonal projectors onto tangent spaces to embedded manifolds.

Any Euclidean space is equipped with an inner product, and from Remark 2.5 we know that the tangent space to an embedded submanifold is a linear subspace of the embedding space $\mathcal{E}$. Therefore it is convenient to equip tangent spaces of embedded submanifolds with the inner product of $\mathcal{E}$, restricting it to the elements of the tangent space. This allows us to define *orthogonal projectors*:

Definition 2.14. *Let $\mathcal{M}$ be an embedded submanifold of a Euclidean space $\mathcal{E}$ equipped with the inner product $\langle\cdot,\cdot\rangle$. A linear operator*

$$\widetilde{\Pi}_x : \mathcal{E} \to \mathcal{T}_x\mathcal{M} \subseteq \mathcal{E}$$

with $\widetilde{\Pi}_x \circ \widetilde{\Pi}_x = \widetilde{\Pi}_x$ is a **projector**. *It is* **orthogonal**, *if*

$$\langle u - \widetilde{\Pi}_x u, v\rangle = 0$$

holds for any $u \in \mathcal{E}$ and $v \in \mathcal{T}_x\mathcal{M}$.

In some cases it is possible to construct such projectors by looking at the *normal space* to some manifold:

Definition 2.15. *The* **normal space** *to an embedded manifold $\mathcal{M}$ is the orthogonal complement (with respect to the induced metric) of the tangent space to $\mathcal{M}$,*

$$\mathcal{N}_x\mathcal{M} = (\mathcal{T}_x\mathcal{M})^\perp = \{u \in \mathcal{E} \mid \langle u, v\rangle = 0 \text{ for all } v \in \mathcal{T}_x\mathcal{M}\} \subseteq \mathcal{E}.$$

Example 2.16 (*Orthogonal projector onto $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$*)**.** We equip $\mathcal{M}_r$ with the Frobenius inner product introduced in Section 2.1. Recall that any $\mathbf{Y} \in \mathcal{M}_r$ has a decomposition into factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ like

$$\mathbf{Y} = \mathbf{U}\mathbf{S}\mathbf{V}^H, \qquad \mathbf{U} \in \mathcal{V}_{m,r}, \quad \mathbf{V} \in \mathcal{V}_{n,r}, \quad \mathbf{S} \in \mathbb{C}^{r\times r} \text{ invertible},$$

see also (2.5).

To determine the orthogonal projector, we follow the approach presented in [Boumal, 2020, Section 7.5], which is based on the normal space of $\mathcal{M}_r$ at $\mathbf{Y}$. Here, we make use of the third representation of $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ given in (2.16). We conclude

$$\mathcal{N}_{\mathbf{Y}}\mathcal{M}_r = \{\mathbf{U}_\perp W \mathbf{V}_\perp^H \mid W \in \mathbb{C}^{(m-r)\times(n-r)} \text{ arbitrary}\}.$$

The orthogonal projection of some $Z \in \mathbb{C}^{m\times n}$ onto $\mathcal{T}_{\mathbf{Y}}\mathcal{M}_r$ therefore satisfies both

$$\widetilde{\Pi}_{\mathbf{Y}} Z = \mathbf{U}\mathbf{M}\mathbf{V}^H + \widetilde{\mathbf{U}}\mathbf{V}^H + \mathbf{U}\widetilde{\mathbf{V}}^H, \tag{2.18}$$

see (2.14), and

$$Z - \widetilde{\Pi}_{\mathbf{Y}} Z = \mathbf{U}_\perp W \mathbf{V}_\perp^H$$

for some W. Adding these equations yields

$$Z = \mathbf{U}\mathbf{M}\mathbf{V}^H + \widetilde{\mathbf{U}}\mathbf{V}^H + \mathbf{U}\widetilde{\mathbf{V}}^H + \mathbf{U}_\perp W \mathbf{V}_\perp^H. \tag{2.19}$$

We define

$$\Pi_{\mathbf{U}} = \mathbf{U}\mathbf{U}^H, \quad \Pi_{\mathbf{U}}^\perp = I_m - \Pi_{\mathbf{U}}, \qquad \Pi_{\mathbf{V}} = \mathbf{V}\mathbf{V}^H, \quad \Pi_{\mathbf{V}}^\perp = I_n - \Pi_{\mathbf{V}}. \tag{2.20}$$

Clearly, due to $\mathbf{U} \in \mathcal{V}_{m,r}$ and $\mathbf{V} \in \mathcal{V}_{n,r}$, it holds that

$$\Pi_{\mathbf{U}}\Pi_{\mathbf{U}} = \Pi_{\mathbf{U}}, \quad \Pi_{\mathbf{V}}\Pi_{\mathbf{V}} = \Pi_{\mathbf{V}} \qquad \text{as well as} \qquad \Pi_{\mathbf{U}}\Pi_{\mathbf{U}}^\perp = 0, \quad \Pi_{\mathbf{V}}\Pi_{\mathbf{V}}^\perp = 0.$$

The matrices $\Pi_{\mathbf{U}}$ and $\Pi_{\mathbf{V}}$ are orthogonal projectors onto the column spaces of $\mathbf{U}$ and $\mathbf{V}$, while $\Pi_{\mathbf{U}}^\perp$ and $\Pi_{\mathbf{V}}^\perp$ are the projectors onto the respective orthogonal complements. From

$$\begin{aligned} \mathbf{U}^H\mathbf{U} &= I_r, &\quad \mathbf{U}^H\widetilde{\mathbf{U}} &= 0, &\quad \mathbf{U}^H\mathbf{U}_\perp &= 0, \\ \mathbf{V}^H\mathbf{V} &= I_r, &\quad \mathbf{V}^H\widetilde{\mathbf{V}} &= 0, &\quad \mathbf{V}^H\mathbf{V}_\perp &= 0, \end{aligned}$$

and (2.19) we obtain the identities

$$\Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}} = \mathbf{U}\mathbf{M}\mathbf{V}^H, \qquad \Pi_{\mathbf{U}}^\perp Z \Pi_{\mathbf{V}} = \widetilde{\mathbf{U}}\mathbf{V}^H, \qquad \Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}}^\perp = \mathbf{U}\widetilde{\mathbf{V}}^H.$$

Plugging these back into (2.18) results in

$$\begin{aligned} \widetilde{\Pi}_{\mathbf{Y}} Z &= \Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}} + \Pi_{\mathbf{U}}^\perp Z \Pi_{\mathbf{V}} + \Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}}^\perp \\ &= \Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}} + Z\Pi_{\mathbf{V}} - \Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}} + \Pi_{\mathbf{U}} Z - \Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}} \\ &= Z\Pi_{\mathbf{V}} - \Pi_{\mathbf{U}} Z \Pi_{\mathbf{V}} + \Pi_{\mathbf{U}} Z. \end{aligned} \tag{2.21}$$

$\circ$

CHAPTER 3

Dynamical low-rank approximation for first-order matrix differential equations

This chapter is concerned with the solution of first-order matrix differential equations of form

$$A'(t) = F\big(A(t)\big), \quad t \in [0, T], \qquad A(0) = A_0 \in \mathbb{C}^{m\times n}. \tag{3.1}$$

Numerous integrators have been constructed for computing approximations to the exact solution $A(t)$ of (3.1). Many of them are tailored to the right-hand side of the problem or desired (geometric) properties of the approximations. An extensive collection of such methods can be found, e.g., in [Hairer et al., 2006, 1993; Hairer and Wanner, 2010].

Here, we are interested in computing approximations $\mathbf{A} \approx A$ which satisfy the rank constraint

$$\operatorname{rank} \mathbf{A} = r \ll \min\{m, n\}.$$

In other words, we search for approximations to the exact solution of (3.1) which are contained in the low-rank manifold $\mathcal{M}_r$ introduced in Section 2.2. Clearly this is only reasonable if the exact solution can be well approximated by a matrix of small rank r. For this chapter we therefore assume that this is indeed the case.

In the following, we first consider differential equations on manifolds. Afterwards we introduce the possibly most fundamental concept for the purposes of this thesis, namely the *dynamical low-rank approximation.* This is followed by an overview of some variants of the original methods proposed throughout the recent years.

3.1 Differential equations on embedded submanifolds

As a starting point, we first discuss first-order differential equations, whose solutions are naturally contained in an embedded submanifold $\mathcal{M}$ of the Euclidean space.

Definition 3.1. *Let $\mathcal{M}$ be an embedded submanifold of $\mathcal{E} = \mathbb{R}^d$. We call the first-order differential equation*

$$a'(t) = f\big(a(t)\big), \quad t \in [0,T], \qquad a(0) = a_0, \tag{3.2}$$

a **differential equation on the manifold $\mathcal{M}$**, *if*

$$a_0 \in \mathcal{M} \quad \textit{implies} \quad a(t) \in \mathcal{M} \textit{ for all } t \in [0,T].$$

The next theorem formulates a criterion for differential equations on a manifold.

Theorem 3.2 ([Hairer et al., 2006, p. 115, Theorem 5.2])**.** *Let $\mathcal{M}$ be an embedded submanifold of the Euclidean space $\mathcal{E}$. The first-order differential equation $a' = f(a)$ as in* Definition 3.1 *is a differential equation on $\mathcal{M}$ if and only if*

$$f(a) \in \mathcal{T}_a\mathcal{M} \quad \textit{for all } a \in \mathcal{M}.$$

Example 3.3. Consider the first-order matrix differential equation

$$A'(t) = LA(t), \quad t \in [0,T], \qquad A(0) = \mathbf{A}_0 \in \mathcal{M}_r, \tag{3.3}$$

where $L \in \mathbb{C}^{m\times m}$. We evaluate the right-hand side at some element $\mathbf{A} \in \mathcal{M}_r$ with the factorization $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H$, and project the result onto the tangent space $\mathcal{T}_\mathbf{A}\mathcal{M}_r$ by applying the orthogonal projector given in (2.21). The orthogonal projector $\widetilde{\Pi}_\mathbf{A}$ onto $\mathcal{T}_\mathbf{A}\mathcal{M}_r$ satisfies

$$\widetilde{\Pi}_\mathbf{A}(L\mathbf{A}) = L\mathbf{A}\Pi_\mathbf{V} - \Pi_\mathbf{U} L\mathbf{A}\Pi_\mathbf{V} + \Pi_\mathbf{U} L\mathbf{A} = L\mathbf{A}\mathbf{V}\mathbf{V}^H - \mathbf{U}\mathbf{U}^H L\mathbf{A}\mathbf{V}\mathbf{V}^H + \mathbf{U}\mathbf{U}^H L\mathbf{A} = L\mathbf{A},$$

and thus $L\mathbf{A} \in \mathcal{T}_\mathbf{A}\mathcal{M}_r$. We conclude from Theorem 3.2 that $A(t) \in \mathcal{M}_r$ for all $t \in [0,T]$.

For this particular right-hand side we are able to obtain the same result directly from Definition 3.1. The exact solution of (3.3) is

$$A(t) = \exp(tL)\mathbf{A}_0.$$

It is well known, that $\exp(tL)$ is invertible for all $t \in \mathbb{R}, L \in \mathbb{C}^{m\times m}$, cf. [Hall, 2015, Proposition 2.3]. Since multiplication with a full-rank matrix preserves the rank, it holds that

$$\operatorname{rank} A(t) = \operatorname{rank}\big(\exp(tL)\mathbf{A}_0\big) = \operatorname{rank}\mathbf{A}_0 = r,$$

thus $A(t) \in \mathcal{M}_r$ for all $t \in [0,T]$. ○

Theorem 3.2 has an important impact on the choice of a numerical method when approximating solutions of differential equations of type (3.2). If the unknown solution is contained in the manifold $\mathcal{M}$, it is favorable to compute numerical approximations which are also contained in $\mathcal{M}$. Standard schemes like Runge-Kutta methods do not yield such approximations directly. The *standard projection method* given in Algorithm 1, [cf. Hairer et al., 2006, p. 110, Algorithm 4.2], yields approximations contained in $\mathcal{M} = \mathcal{M}_r$. The orthogonal projector Π which computes the low-rank matrix $\mathbf{A} = \Pi A$ is implicitly characterized by the best-approximation condition

$$\mathbf{A} = \operatorname*{argmin}_{\mathbf{Y}\in\mathcal{M}_r} \|\mathbf{Y} - A\|. \tag{3.4}$$

The solution $\mathbf{A} \in \mathcal{M}_r$ can be computed by a truncated singular value decomposition, where one only keeps the r largest singular values and omits all others, cf. Theorem A.8.

Algorithm 1: Standard projection method

```
Assume that $\mathbf{A}_k \in \mathcal{M}$. One step $\mathbf{A}_k \mapsto \mathbf{A}_{k+1} \in \mathcal{M}$ is performed like
1 Compute one step of a numerical integrator with initial $\mathbf{A}_k$, yielding $A_{k+1}$.
2 Project $A_{k+1}$ back onto the manifold with a suitable projection, obtain $\mathbf{A}_{k+1} \in \mathcal{M}$.
```

We now return to the problem of finding a low-rank approximation to the exact solution $A(t)$ of (3.1). For small dimensions, we employ the standard projection method, where we compute approximations $A_k \approx A(t_k)$ with a standard numerical scheme, and then compute the rank-r best-approximation $\mathbf{A}_k$ to A_k by the truncated singular value decomposition afterwards. However, if the dimension of the problem is large, the truncated singular value decomposition is expensive and the standard projection method is not practical anymore.

3.2 Dynamical low-rank approximation

A cheap alternative to the standard projection method was introduced in [Koch and Lubich, 2007], the so-called *dynamical low-rank approximation*. The basic concepts have been formulated for the simpler problem of determining a low-rank approximation dynamically to a given, time-dependent matrix $A(t) \in \mathbb{C}^{m\times n}$. The approximations are contained in the low-rank manifold permanently, so that no subsequent projection onto $\mathcal{M}_r$ is required.

The core idea of the new ansatz is to replace (3.4) by

$$\|\mathbf{A}'(t) - A'(t)\| \overset{!}{=} \min. \tag{3.5}$$

If $\mathbf{A}(t)$ is contained in the low-rank manifold $\mathcal{M}_r$, its derivative is an element of the tangent space $\mathcal{T}_{\mathbf{A}(t)}\mathcal{M}_r$ to $\mathcal{M}_r$ at $\mathbf{A}(t)$, see Corollary 2.13.

To get an idea of the situation, recall that $\mathcal{M}_r$ is an embedded submanifold of the Euclidean space. In Figure 3.1, the derivatives of A and $\mathbf{A}$ are displayed as vectors in the embedding Euclidean space. For given A', $\|\mathbf{A}' - A'\|$ becomes minimal if the vector $\mathbf{A}' - A'$ is perpendicular to the tangent space $\mathcal{T}_{\mathbf{A}(t)}\mathcal{M}_r$. In other words, (3.5) is equivalent to the following Galerkin condition on $\mathcal{T}_{\mathbf{A}(t)}\mathcal{M}_r$ (we omit the argument t): find $\mathbf{A}' \in \mathcal{T}_{\mathbf{A}}\mathcal{M}_r$ satisfying

$$\langle \mathbf{A}' - A', \delta\mathbf{A}\rangle = 0 \qquad \text{for all} \quad \delta\mathbf{A} \in \mathcal{T}_{\mathbf{A}}\mathcal{M}_r. \tag{3.6}$$

From this formulation, differential equations for the factors of $\mathbf{A}$ are derived:

Proposition 3.4 ([Koch and Lubich, 2007, Proposition 2.1]). *For the map $\mathbf{A} : [0, T] \to \mathcal{M}_r$, $t \mapsto \mathbf{A}(t) = \mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H \in \mathcal{M}_r$ with nonsingular $\mathbf{S} \in \mathbb{C}^{r\times r}$ and with $\mathbf{U} \in \mathcal{V}_{m,r}$, $\mathbf{V} \in \mathcal{V}_{n,r}$, condition (3.6) is equivalent to $\mathbf{A}' = \mathbf{U}'\mathbf{S}\mathbf{V}^H + \mathbf{U}\mathbf{S}'\mathbf{V}^H + \mathbf{U}\mathbf{S}(\mathbf{V}')^H$, where*

$$\begin{aligned} \mathbf{S}' &= \mathbf{U}^H A'\mathbf{V}, \\ \mathbf{U}' &= \Pi^{\perp}_{\mathbf{U}} A'\mathbf{V}\mathbf{S}^{-1}, \\ \mathbf{V}' &= \Pi^{\perp}_{\mathbf{V}} (A')^H\mathbf{U}\mathbf{S}^{-H}, \end{aligned} \tag{3.7}$$

with the orthogonal projections $\Pi^{\perp}_{\mathbf{U}}, \Pi^{\perp}_{\mathbf{V}}$ as given in (2.20).

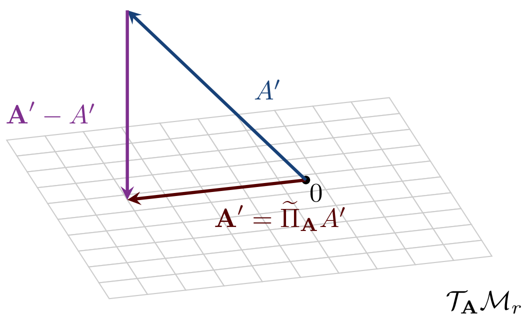
$\mathbf{A}' - A'$
A'
0
$\mathbf{A}' = \widetilde{\Pi}_\mathbf{A} A'$
$\mathcal{T}_\mathbf{A}\mathcal{M}_r$

Figure 3.1. Graphical illustration of the Galerkin condition (3.6). Only for $\mathbf{A}' - A'$ perpendicular to $\mathcal{T}_\mathbf{A}\mathcal{M}_r$ the minimizing condition (3.5) is satisfied. For given A', this is only the case for $\mathbf{A}' = \widetilde{\Pi}_\mathbf{A} A'$ with $\widetilde{\Pi}_\mathbf{A}$ from (2.21).

The differential equations (3.7) may be solved numerically with a standard integration scheme, e.g., implicit Runge-Kutta methods. Unfortunately, it was shown that the proposed ansatz suffers from *overapproximation*, i.e., if $r > \operatorname{rank} A$. Then singular values with value zero are introduced in the approximation $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H$, which causes the factor $\mathbf{S}$ to be singular. Since the inverse of $\mathbf{S}$ is required in the updates for the factors $\mathbf{U}$ and $\mathbf{V}$, matrices $\mathbf{S}$ with tiny singular values turn the computation highly unstable.

In [Lubich and Oseledets, 2014], the authors propose a different ansatz for dynamical low-rank approximation. It is also based on the Galerkin condition (3.6), but does not suffer from overapproximation. The solution of the minimization problem can be given in terms of the orthogonal projector $\widetilde{\Pi}_\mathbf{A}$ onto $\mathcal{T}_\mathbf{A}\mathcal{M}_r$, cf. (2.21), see also Figure 3.1:

$$\mathbf{A}' = \widetilde{\Pi}_\mathbf{A} A' = A'\Pi_\mathbf{V} - \Pi_\mathbf{U} A' \Pi_\mathbf{V} + \Pi_\mathbf{U} A'. \tag{3.8}$$

Combined with an initial value $\mathbf{A}_0 \in \mathcal{M}_r$, typically determined as the rank-r best-approximation to A_0, this yields an evolution equation for $\mathbf{A}(t)$ whose solution is contained in $\mathcal{M}_r$ for all times t due to Theorem 3.2. Unfortunately, the projected differential equation cannot be solved exactly in general. As a remedy, the authors suggest to perform a Lie-Trotter splitting with step size $\tau > 0$ on the right-hand side of (3.8) (for a short note on splitting methods we refer to Appendix B). This yields the three subproblems

$$\begin{aligned}
\mathbf{A}_\mathrm{I}' &= A'\Pi_{\mathbf{V}_\mathrm{I}} && \text{on} \quad [0,\tau], && \mathbf{A}_\mathrm{I}(0) = \mathbf{A}_0,\\
\mathbf{A}_\mathrm{II}' &= -\Pi_{\mathbf{U}_\mathrm{II}} A' \Pi_{\mathbf{V}_\mathrm{II}} && \text{on} \quad [0,\tau], && \mathbf{A}_\mathrm{II}(0) = \mathbf{A}_\mathrm{I}(\tau),\\
\mathbf{A}_\mathrm{III}' &= \Pi_{\mathbf{U}_\mathrm{III}} A' && \text{on} \quad [0,\tau], && \mathbf{A}_\mathrm{III}(0) = \mathbf{A}_\mathrm{II}(\tau).
\end{aligned} \tag{3.9}$$

All subproblems in (3.9) can be solved exactly on the low-rank manifold $\mathcal{M}_r$ if suitable additional conditions on the derivatives of $\mathbf{U}$ and $\mathbf{V}$ are imposed, cf. [Lubich and Oseledets, 2014, Lemma 3.1]:

$$\begin{aligned}
\mathbf{A}_\mathrm{I}(t) &= \mathbf{U}_\mathrm{I}(t)\mathbf{S}_\mathrm{I}(t)\mathbf{V}_\mathrm{I}(t)^H, && \text{with} && (\mathbf{U}_\mathrm{I}\mathbf{S}_\mathrm{I})' = A'\mathbf{V}_\mathrm{I}, && \mathbf{V}_\mathrm{I}' = 0,\\
\mathbf{A}_\mathrm{II}(t) &= \mathbf{U}_\mathrm{II}(t)\mathbf{S}_\mathrm{II}(t)\mathbf{V}_\mathrm{II}(t)^H, && \text{with} && \mathbf{S}_\mathrm{II}' = -\mathbf{U}_\mathrm{II}^H A'\mathbf{V}_\mathrm{II}, && \mathbf{U}_\mathrm{II}' = 0, \quad \mathbf{V}_\mathrm{II}' = 0,\\
\mathbf{A}_\mathrm{III}(t) &= \mathbf{U}_\mathrm{III}(t)\mathbf{S}_\mathrm{III}(t)\mathbf{V}_\mathrm{III}(t)^H, && \text{with} && (\mathbf{V}_\mathrm{III}\mathbf{S}_\mathrm{III}^H)' = (A')^H\mathbf{U}_\mathrm{III}, && \mathbf{U}_\mathrm{III}' = 0.
\end{aligned}$$

Then, the solutions read

$$\begin{aligned}\mathbf{A}_{\mathrm{I}}(t) &= \big(\mathbf{U}_{\mathrm{I}}(0)\mathbf{S}_{\mathrm{I}}(0) + (A(t) - A(0))\mathbf{V}_{\mathrm{I}}(0)\big)\mathbf{V}_{\mathrm{I}}(0)^H,\\ \mathbf{A}_{\mathrm{II}}(t) &= \mathbf{U}_{\mathrm{II}}(0)\big(\mathbf{S}_{\mathrm{II}}(0) - \mathbf{U}_{\mathrm{II}}(0)^H(A(t) - A(0))\mathbf{V}_{\mathrm{II}}(0)\big)\mathbf{V}_{\mathrm{II}}(0)^H,\\ \mathbf{A}_{\mathrm{III}}(t) &= \mathbf{U}_{\mathrm{III}}(0)\big(\mathbf{S}_{\mathrm{III}}(0)\mathbf{V}_{\mathrm{III}}(0)^H + \mathbf{U}_{\mathrm{III}}(0)^H(A(t) - A(0))\big).\end{aligned}$$

Finally, one obtains $\mathbf{A}_1 = \mathbf{A}_{\mathrm{III}}(\tau)$ as a low-rank approximation to $A(\tau)$.

Overall, given a low-rank decomposition of $\mathbf{A}_0 = \mathbf{U}_0\mathbf{S}_0\mathbf{V}_0^H \approx A(0)$ and introducing the increment $\Delta A = A(\tau) - A(0)$, this yields the three update steps

$$\begin{aligned}\mathbf{U}_1\widehat{\mathbf{S}}_1 &= \mathbf{U}_0\mathbf{S}_0 + \Delta A\mathbf{V}_0,\\ \widetilde{\mathbf{S}}_0 &= \widehat{\mathbf{S}}_1 - \mathbf{U}_1^H\Delta A\mathbf{V}_0,\\ \mathbf{V}_1\mathbf{S}_1^H &= \mathbf{V}_0\widetilde{\mathbf{S}}_0^H + \Delta A^H\mathbf{U}_1,\end{aligned} \tag{3.10}$$

which allow to compute the approximation $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H = \mathbf{A}_{\mathrm{III}}(\tau) \approx A(\tau)$ dynamically, see also Algorithm 2.

Remark 3.5. Note that in all three update steps in (3.10) matrices of column dimension r are computed. It is thus beneficial to implement the updates such, that also in all intermediate calculations the column dimension of all contributing matrices does not exceed r. This requires special attention on the implementation of the projector-splitting integrator. ◇

To continue the integration, one uses $\mathbf{A}_1$ as initial value for the next integration step and the new increment $\Delta A = A(2\tau) - A(\tau)$, and so on. Repeating this k times gives an approximation $\mathbf{A}_k \approx A(k\tau)$. The numerical method has been named *projector-splitting integrator* in [Lubich and Oseledets, 2014].

Algorithm 2: Projector-splitting integrator for low-rank approximations to given time-dependent matrices [Lubich and Oseledets, 2014, Section 3.2], single time step

```
1 prsi($\mathbf{U}, \mathbf{S}, \mathbf{V}, r, \Delta A$)
  Input : factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of rank-$r$ approximation $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H \approx A(t)$ with $\mathbf{U} \in \mathcal{V}_{m,r}$, $\mathbf{V} \in \mathcal{V}_{n,r}$,
          $\mathbf{S} \in \mathbb{C}^{r\times r}$, increment $\Delta A = A(t+\tau) - A(t)$
2 $\widetilde{\mathbf{K}} = \Delta A\mathbf{V}$
3 $\mathbf{K} = \mathbf{U}\mathbf{S} + \widetilde{\mathbf{K}}$
4 compute reduced $QR$-decomposition $\mathbf{U}_1\widehat{\mathbf{S}}_1 = \mathbf{K}$
5 $\widetilde{\mathbf{S}}_0 = \widehat{\mathbf{S}}_1 - \mathbf{U}_1^H\widetilde{\mathbf{K}}$
6 $\mathbf{L} = \mathbf{V}\widetilde{\mathbf{S}}_0^H + \Delta A^H\mathbf{U}_1$
7 compute reduced $QR$-decomposition $\mathbf{V}_1\mathbf{S}_1^H = \mathbf{L}$
8 Return $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1, \mathbf{L}$
  Output: factors $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1$ of rank-$r$ approximation $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H \approx A(t+\tau)$ and $\mathbf{L} = \mathbf{V}_1\mathbf{S}_1^H$
          with $\mathbf{U}_1 \in \mathcal{V}_{m,r}$, $\mathbf{V}_1 \in \mathcal{V}_{n,r}$, $\mathbf{S}_1 \in \mathbb{C}^{r\times r}$
```

In [Koch and Lubich, 2007], the authors list several reasons, why the technique of dynamical low-rank approximation is superior to the standard projection method. Most important for us, the ansatz extends straightforwardly to computing a low-rank approximation to the unknown solution of a first-order matrix differential equation of type (3.1). Substituting (3.1) into (3.8) yields

$$\mathbf{A}' = \widetilde{\Pi}_{\mathbf{A}} F(A).$$

Unless the exact solution is known, A is replaced by its low-rank approximation $\mathbf{A}$ in F to obtain

$$\mathbf{A}' = \widetilde{\Pi}_{\mathbf{A}} F(\mathbf{A}), \qquad \mathbf{A}(0) = \mathbf{A}_0 \approx A_0. \tag{3.11}$$

Again, using the representation of the projector and performing a Lie-Trotter splitting gives rise to three subproblems. If these can be solved exactly, one can formulate Algorithm 3, which is a projector-splitting integrator for computing low-rank approximations to the unknown solution of (3.1), cf. [Ceruti and Lubich, 2021, Section 2]. Note that the update step for the $\mathbf{S}$ matrix can be interpreted as a backwards step in time due to the negative sign in the right-hand side of the respective subproblem, while the other two updates can be seen as forward steps.

Algorithm 3: Projector-splitting integrator for low-rank approximations to unknown solution $A(t)$ to (3.1), single time step

```
1 prsiF($\mathbf{U}, \mathbf{S}, \mathbf{V}, r, \tau, F$)
  Input : factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of rank-$r$ approximation $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H \approx A(t)$ with $\mathbf{U} \in \mathcal{V}_{m,r}$, $\mathbf{V} \in \mathcal{V}_{n,r}$,
          $\mathbf{S} \in \mathbb{C}^{r\times r}$, step size $\tau$, right-hand side $F$
2 Solve
      $\mathbf{K}'(t) = F\big(\mathbf{K}(t)\mathbf{V}^H\big)\mathbf{V}, \quad \mathbf{K}(0) = \mathbf{U}\mathbf{S},$
  for $t \in [0, \tau]$ and compute a $QR$-decomposition $\mathbf{U}_1\widehat{\mathbf{S}}_1 = \mathbf{K}(\tau)$.
3 Solve
      $\mathbf{S}'(t) = -\mathbf{U}_1^H F\big(\mathbf{U}_1\mathbf{S}(t)\mathbf{V}^H\big)\mathbf{V}, \quad \mathbf{S}(0) = \widehat{\mathbf{S}}_1,$
  for $t \in [0, \tau]$ and set $\widetilde{\mathbf{S}}_0 = \mathbf{S}(\tau)$.
4 Solve
      $\mathbf{L}'(t) = F\big(\mathbf{U}_1\mathbf{L}(t)^H\big)^H\mathbf{U}_1, \quad \mathbf{L}(0) = \mathbf{V}\widetilde{\mathbf{S}}_0^H,$
  for $t \in [0, \tau]$ and compute a $QR$-decomposition $\mathbf{V}_1\mathbf{S}_1^H = \mathbf{L}_1 = \mathbf{L}(\tau)$.
5 Return $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1, \mathbf{L}_1$
  Output: factors $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1$ of rank-$r$ approximation $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H \approx A(t+\tau)$ and $\mathbf{L}_1 = \mathbf{V}_1\mathbf{S}_1^H$
          with $\mathbf{U}_1 \in \mathcal{V}_{m,r}$, $\mathbf{V}_1 \in \mathcal{V}_{n,r}$, $\mathbf{S}_1 \in \mathbb{C}^{r\times r}$
```

In general, the subproblems in Algorithm 3 cannot be solved exactly. Then it is necessary to employ a numerical method for solving the differential equations for $\mathbf{K}, \mathbf{S}$, and $\mathbf{L}$ in Algorithm 3 approximately, e.g., a Runge-Kutta method. The simplest choice is the explicit Euler method of order 1 with step size τ. In contrast to Algorithm 2, it is not possible to reuse quantities from previous substeps of the scheme. Instead, the right-hand side F needs to be evaluated at the current low-rank approximation and hence three times per step. For a Runge-Kutta method with s internal stages, the right-hand side obviously needs to be evaluated $3s$ times.

Unlike the dynamical low-rank integrator constructed from Proposition 3.4, the projector-splitting integrator as in Algorithm 3 has the favorable property of being robust with respect to the presence of small singular values of the solution A to (3.1) or its approximation $\mathbf{A}$. This also includes the case of overapproximation, where the low-rank factor $\mathbf{S}$ is not invertible.

Theorem 3.6 (Robust error bound, [Kieri et al., 2016, Theorem 2.1]). *Let $A(t)$ denote the solution of (3.1). Assume the following conditions:*

1. *F is Lipschitz-continuous and bounded: there exist $L, B > 0$ such that for all $Y, \widetilde{Y} \in \mathbb{C}^{m\times n}$ and $0 \le t \le T$ it holds that*
$$\|F(Y) - F(\widetilde{Y})\| \le L\|Y - \widetilde{Y}\|, \qquad \|F(Y)\| \le B.$$
2. *$F(\mathbf{A})$ is in the tangent space $\mathcal{T}_{\mathbf{A}}\mathcal{M}_r$ up to a small remainder,*
$$\|\widetilde{\Pi}_{\mathbf{A}}^{\perp} F(\mathbf{A})\| \le \varepsilon$$
for all $\mathbf{A} \in \mathcal{M}_r$ in a neighborhood of $A(t)$ and $0 \le t \le T$.
3. *The initial value $A(0)$ and the starting value $\mathbf{A}_0 \in \mathcal{M}_r$ are δ-close,*
$$\|A(0) - \mathbf{A}_0\| \le \delta.$$

Denote by $\mathbf{A}_k$ the rank-r approximation to $A(t_k)$ obtained after k steps of Algorithm 3 and step size $\tau > 0$. Then, the error satisfies for all k with $t_k \le T$
$$\|A(t_k) - \mathbf{A}_k\| \le c_0\delta + c_1\varepsilon + c_2\tau,$$
where for an upper bound $\tau_0 \ge \tau$ on the step size the constants are explicitly given as
$$c_0 = \mathrm{e}^{LT}, \qquad c_1 = (4 + 3\,\mathrm{e}^{L\tau_0})\frac{\mathrm{e}^{LT} - 1}{L}, \qquad c_2 = (9 + 4\,\mathrm{e}^{L\tau_0}\, B\,\mathrm{e}^{LT} - 1),$$
and hence are independent of the singular values of the exact or approximate solution.

When the subproblems in Algorithm 3 are not solved exactly but approximately with a numerical method, an additional error is introduced. This error is bounded in terms of the local errors in the respective inexact substeps, where the constants are independent of small singular values, cf. [Kieri et al., 2016, Section 2.6.3].

A similar result for the projector-splitting integrator with increments, cf. Algorithm 2, is obtained as a byproduct from the above. Interestingly, the error bound becomes independent of the step size τ:

Theorem 3.7 ([Kieri et al., 2016, Section 2.6.1]). *Let*
$$A(t) = \mathbf{X}(t) + R(t) \qquad \text{with} \quad \mathbf{X}(t) \in \mathcal{M}_r, \quad \|R(0)\| \le \delta, \quad \|R'(t)\| \le \varepsilon.$$
Then the error between $A(t_k)$ and the low-rank approximation $\mathbf{A}_k \approx A(t_k)$ computed by performing k steps with Algorithm 2 is bounded by
$$\|A(t_k) - \mathbf{A}_k\| \le \delta + 7t_k\varepsilon.$$

Note that if $R(t) = 0$ for all t, i.e., $A(t) \in \mathcal{M}_r$ is in fact for all t a rank-r matrix, then $\delta = \varepsilon = 0$ and Algorithm 2 computes $\mathbf{A}_k = A(t_k)$ exactly. This favorable property is also called the *exactness property* and was first proven in [Lubich and Oseledets, 2014, Theorem 4.1].

Remark 3.8. Permuting the order of the substeps in the Lie-Trotter splitting yields alternative algorithms for the dynamical low-rank approximation of the solution to (3.1). However, the exactness property is only satisfied for the ordering as in (3.9) or by interchanging steps I and III. The proof of the exactness property for the latter case is done similarly to the proof in [Lubich and Oseledets, 2014]. ◇

3.3 Dynamical low-rank approximation for stiff first-order matrix differential equations

Consider the semilinear first-order equation

$$A'(t) = L_1 A(t) + A(t) L_2 + f\big(A(t)\big), \quad t \in [0,T], \qquad A(0) = A_0 \in \mathbb{C}^{m\times n}, \tag{3.12}$$

where $L_1 \in \mathbb{C}^{m\times m}$ and $L_2 \in \mathbb{C}^{n\times n}$ are constant matrices and f is a nonlinearity. Such equations typically originate from the space discretization of PDEs. In this context, L_1 and L_2 are discretized counterparts of differential operators. When the norms of L_1 and L_2 become large for fine grids, (3.12) becomes a stiff matrix differential equation. Explicit numerical integrators such as the projector-splitting integrator usually suffer from severe step size restrictions when they are applied to such equations.

In [Ostermann et al., 2019], the special case $L_2 = L_1^H$ was considered. It was proposed to split the right-hand side of (3.12) into the stiff linear part, $L_1 A(t) + A(t) L_2$, and the nonstiff nonlinear part, $f\big(A(t)\big)$, and to apply a Lie-Trotter splitting. This yields a method of order 1. Solving the subproblems exactly or with some standard numerical scheme does not necessarily preserve the rank of the initial value A_0. Hence, the resulting method might yield a full-rank solution. A low-rank integrator for first-order equations of the type (3.12) is obtained by solving

$$A'_{\text{lin}}(t) = L_1 A_{\text{lin}}(t) + A_{\text{lin}}(t) L_2, \tag{3.13a}$$

$$A'_{\text{non}}(t) = f\big(A_{\text{non}}(t)\big), \tag{3.13b}$$

on the low-rank manifold $\mathcal{M}_r$. For the linear subproblem, with an initial value $\mathbf{A}_0 \in \mathcal{M}_r$, the solutions of the linear subproblem are also in $\mathcal{M}_r$:

Lemma 3.9. *Let $\mathbf{A}_0 \in \mathcal{M}_r$ be the initial value for the linear subproblem (3.13a). Then the solution of (3.13a) is also in $\mathcal{M}_r$ for all $t > 0$.*

Proof. We first show, that the right-hand side of (3.13a) evaluated at $\mathbf{A} \in \mathcal{M}_r$ remains invariant under orthogonal projection onto $\mathcal{T}_{\mathbf{A}}\mathcal{M}_r$ with $\widetilde{\Pi}_{\mathbf{A}}$ given in (2.21):

$$\begin{aligned}\widetilde{\Pi}_{\mathbf{A}}(L_1\mathbf{A} + \mathbf{A}L_2) &= L_1\mathbf{A}\mathbf{V}\mathbf{V}^H - \mathbf{U}\mathbf{U}^H L_1 \mathbf{A}\mathbf{V}\mathbf{V}^H + \mathbf{U}\mathbf{U}^H L_1\mathbf{A} + \mathbf{A}L_2\mathbf{V}\mathbf{V}^H - \mathbf{U}\mathbf{U}^H\mathbf{A}L_2\mathbf{V}\mathbf{V}^H \\ &\quad + \mathbf{U}\mathbf{U}^H\mathbf{A}L_2 \\ &= L_1\mathbf{A} - \mathbf{U}\mathbf{U}^H L_1\mathbf{A} + \mathbf{U}\mathbf{U}^H L_1\mathbf{A} + \mathbf{A}L_2\mathbf{V}\mathbf{V}^H - \mathbf{A}L_2\mathbf{V}\mathbf{V}^H + \mathbf{A}L_2 \\ &= L_1\mathbf{A} + \mathbf{A}L_2.\end{aligned}$$

From Theorem 3.2 we conclude that the exact solution $\mathbf{A}_{\text{lin}}(t)$ of

$$\mathbf{A}'_{\text{lin}}(t) = L_1\mathbf{A}_{\text{lin}}(t) + \mathbf{A}_{\text{lin}}(t)L_2, \qquad \mathbf{A}_{\text{lin}}(0) = \mathbf{A}^0_{\text{lin}} \in \mathcal{M}_r, \tag{3.14}$$

satisfies $\mathbf{A}_{\text{lin}}(t) \in \mathcal{M}_r$ for all $t \geq 0$. □

The exact solution of (3.14) can be given in terms of the matrix exponential and reads

$$\mathbf{A}_{\text{lin}}(t) = \mathrm{e}^{tL_1}\, \mathbf{A}^0_{\text{lin}}\, \mathrm{e}^{tL_2}, \qquad t \geq 0. \tag{3.15}$$

This is verified by straightforward calculation: Obviously, $\mathbf{A}_{\text{lin}}(0) = \mathbf{A}^0_{\text{lin}}$. Further, by the product rule and Theorem A.21 the derivative of $\mathbf{A}_{\text{lin}}$ is given by

$$\mathbf{A}'_{\text{lin}}(t) = L_1\, \mathrm{e}^{tL_1}\, \mathbf{A}^0_{\text{lin}}\, \mathrm{e}^{tL_2} + \mathrm{e}^{tL_1}\, \mathbf{A}^0_{\text{lin}}\, \mathrm{e}^{tL_2}\, L_2 = L_1 \mathbf{A}_{\text{lin}}(t) + \mathbf{A}_{\text{lin}}(t) L_2.$$

A low-rank approximation $\mathbf{A}_1 \approx A(\tau)$ is obtained by solving the projected second subproblem

$$\mathbf{A}'_{\text{non}}(t) = \widetilde{\Pi}_{\mathbf{A}_{\text{non}}(t)} f\big(\mathbf{A}_{\text{non}}(t)\big), \quad t \in [0, \tau], \qquad \mathbf{A}_{\text{non}}(0) = \mathbf{A}_{\text{lin}}(\tau) \in \mathcal{M}_r,$$

approximately with Algorithm 3. Iterating k times yields the approximation $\mathbf{A}_k \approx A(t_k)$. A single time step of the low-rank integrator is presented in Algorithm 4.

Algorithm 4: Projector-splitting integrator for stiff semilinear first-order ODEs (3.12) [Ostermann et al., 2019], single time step

```
prsistiff($\mathbf{U}, \mathbf{S}, \mathbf{V}, r, \tau, L_1, L_2, f$)
Input : factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of rank-$r$ approximation $\mathbf{A} \approx A(t)$ with $\mathbf{U} \in \mathcal{V}_{m,r}$, $\mathbf{V} \in \mathcal{V}_{n,r}$, $\mathbf{S} \in \mathbb{C}^{r\times r}$,
        step size $\tau$, matrices $L_1$ and $L_2$, right-hand side $f$
First subproblem – evaluate the action of the matrix exponential on the factors
3 Compute $QR$-decomposition of $\widetilde{\mathbf{U}}\mathbf{S}_1 = \mathrm{e}^{\tau L_1}\mathbf{U}$, $\mathbf{S}_1$ invertible, $\widetilde{\mathbf{U}}$ orthonormal
4 Compute $QR$-decomposition of $\widetilde{\mathbf{V}}\mathbf{S}_2 = \mathrm{e}^{\tau L_2^H}\mathbf{V}$, $\mathbf{S}_2$ invertible, $\widetilde{\mathbf{V}}$ orthonormal
5 Set $\widetilde{\mathbf{S}} = \mathbf{S}_1\mathbf{S}\mathbf{S}_2^H$
Second subproblem – solve nonlinear subproblem approximately on $\mathcal{M}_r$
7 $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1, \mathbf{L}_1 =$ prsiF($\widetilde{\mathbf{U}}, \widetilde{\mathbf{S}}, \widetilde{\mathbf{V}}, r, \tau, f$)
8 Return $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1, \mathbf{L}_1$
Output: factors $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1$ of rank-$r$ approximation $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H \approx A(t+\tau)$ and $\mathbf{L}_1 = \mathbf{V}_1\mathbf{S}_1^H$
        with $\mathbf{U}_1 \in \mathcal{V}_{m,r}$, $\mathbf{V}_1 \in \mathcal{V}_{n,r}$, $\mathbf{S}_1 \in \mathbb{C}^{r\times r}$
```

3.4 Other dynamical low-rank integrators for first-order problems

3.4.1 Time integration of (skew-)Hermitian low-rank matrices

Next we consider matrix differential equations whose solutions are (skew-)Hermitian. In [Ceruti and Lubich, 2020], a (skew-)symmetry preserving dynamical low-rank integrator was constructed. Though it was proposed only for real matrices, it straightforwardly generalizes to the complex case. The right-hand side F is assumed to satisfy

$$F\big(A(t)\big) \text{ is (skew-)Hermitian whenever } A(t) \in \mathbb{C}^{m\times n} \text{ is (skew-)Hermitian.}$$

Then, the solution of the matrix differential equation (3.1) is (skew-)Hermitian if the initial value is (skew-)Hermitian, and the same holds for the projected differential equation (3.11). Hence, the low-rank approximation $\mathbf{A} \approx A$ admits a decomposition like $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{U}^H$. The modified integrator shares the first substep with the projector-splitting integrator as in Algorithm 3. The second substep however is changed, and due to $\mathbf{V} = \mathbf{U}$ the third substep is not necessary. A single time step of the variant is presented in Algorithm 5. Though the method is derived by non-trivial modifications of the projector-splitting integrator, the authors of [Ceruti and Lubich, 2020] point out that it cannot be interpreted as a splitting integrator.

Algorithm 5: (Skew-)Hermitian preserving integrator [Ceruti and Lubich, 2020, Algorithm 1]

```
1 prsiskew($\mathbf{U}, \mathbf{S}, r, \tau, F$)
  Input : factors $\mathbf{U}, \mathbf{S}$ of rank-$r$ approximation $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{U}^H \approx A(t)$ with $\mathbf{U} \in \mathcal{V}_{m,r}$, $\mathbf{S} \in \mathbb{C}^{r\times r}$,
          step size $\tau$, right-hand side $F$
2 Solve
      $\mathbf{K}'(t) = F\big(\mathbf{K}(t)\mathbf{U}^H\big)\mathbf{U}, \quad \mathbf{K}(0) = \mathbf{U}\mathbf{S},$
  for $t \in [0,\tau]$ and compute a $QR$-decomposition $\mathbf{U}_1\mathbf{R} = \mathbf{K}(\tau)$.
3 Solve
      $\mathbf{S}'(t) = \mathbf{U}_1^H F\big(\mathbf{U}_1\mathbf{S}(t)\mathbf{U}_1^H\big)\mathbf{U}_1, \quad \mathbf{S}(0) = \mathbf{U}_1^H\mathbf{A}_0\mathbf{U}_1$
  for $t \in [0,\tau]$ and set $\mathbf{S}_1 = \mathbf{S}(\tau)$.
4 Return $\mathbf{U}_1, \mathbf{S}_1$
  Output: factors $\mathbf{U}_1, \mathbf{S}_1$ of rank-$r$ approximation $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{U}_1^H \approx A(t+\tau)$ with $\mathbf{U}_1 \in \mathcal{V}_{m,r}$,
          $\mathbf{S}_1 \in \mathbb{C}^{r\times r}$
```

It was shown in [Ceruti and Lubich, 2020, Section 3.2], that both the exactness property and the robust error bound hold verbatim for the new integrator Algorithm 5.

3.4.2 Unconventional robust dynamical low-rank integrator

In [Ceruti and Lubich, 2021], the authors introduced another dynamical low-rank integrator for first-order matrix differential equations (3.1), which was named the *unconventional robust integrator*. It can be viewed as a variant of the projector-splitting integrator, and it is applicable to both the problem of finding low-rank approximations to given, time-dependent matrices and to the unknown solution of a first-order matrix differential equation (3.1). Furthermore, it was shown that also for this integrator the exactness property and the robust error bound Theorem 3.6 are valid.

There are three main differences between the projector-splitting integrator and the unconventional robust integrator, cf. [Ceruti and Lubich, 2021, Section 3]:

1. The projector-splitting integrator is a sequential algorithm, the substeps cannot be carried out in parallel. This is different for the new integrator, where the matrices $\mathbf{K}$ and $\mathbf{L}$ as in Algorithm 2 and Algorithm 3, respectively, can be computed simultaneously.

2. The update step for the matrix $\mathbf{S}$ is now a forward time step, while in the projector-splitting integrator it is a backwards step. Note that this is especially relevant for strongly dissipative problems, where a backwards step might cause stability issues.

3. The unconventional robust integrator preserves (skew-)symmetry, which is not true for the projector-splitting integrator.

A single time step of the unconventional robust integrator is presented in Algorithm 6.

Algorithm 6: Unconventional robust integrator [Ceruti and Lubich, 2021, Section 3.1]

```
1 prsirobust($\mathbf{U}, \mathbf{S}, \mathbf{V}, r, \tau, F$)
  Input  : factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of rank-$r$ approximation $\mathbf{A}_0 = \mathbf{U}\mathbf{S}\mathbf{V}^H \approx A(t)$ with $\mathbf{U} \in \mathcal{V}_{m,r}$, $\mathbf{V} \in \mathcal{V}_{n,r}$,
           $\mathbf{S} \in \mathbb{C}^{r\times r}$, step size $\tau$, right-hand side $F$
2 Solve
      $$\mathbf{K}'(t) = F\big(\mathbf{K}(t)\mathbf{V}^H\big)\mathbf{V}, \quad \mathbf{K}(0) = \mathbf{U}\mathbf{S},$$
  for $t \in [0,\tau]$. Compute a $QR$-decomposition $\mathbf{U}_1\mathbf{R}_1 = \mathbf{K}(\tau)$ and define $\mathbf{M} = \mathbf{U}_1^H\mathbf{U}$.
3 Solve
      $$\mathbf{L}'(t) = F\big(\mathbf{U}\mathbf{L}(t)^H\big)^H\mathbf{U}, \quad \mathbf{L}(0) = \mathbf{V}\mathbf{S}^H,$$
  for $t \in [0,\tau]$. Compute a $QR$-decomposition $\mathbf{V}_1\widetilde{\mathbf{R}}_1 = \mathbf{L}(\tau)$ and define $\mathbf{N} = \mathbf{V}_1^H\mathbf{V}$.
4 Solve
      $$\mathbf{S}'(t) = \mathbf{U}_1^H F\big(\mathbf{U}_1\mathbf{S}(t)\mathbf{V}_1^H\big)^H\mathbf{V}_1, \quad \mathbf{S}(0) = \mathbf{M}\mathbf{S}\mathbf{N}^H,$$
  for $t \in [0,\tau]$ and set $\mathbf{S}_1 = \mathbf{S}(\tau)$.
5 Return $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1$
  Output: factors $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1$ of rank-$r$ approximation $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H \approx A(t+\tau)$ with $\mathbf{U}_1 \in \mathcal{V}_{m,r}$,
          $\mathbf{V}_1 \in \mathcal{V}_{n,r}$, $\mathbf{S}_1 \in \mathbb{C}^{r\times r}$
```

CHAPTER 4

Dynamical low-rank approximation for second-order matrix differential equations

While dynamical low-rank approximation for first-order matrix differential equations has been studied extensively in the past, to the best of our knowledge second-order matrix differential equations have not been considered so far. A prototype of such an equation is given by

$$A''(t) = F\big(A(t)\big) \in \mathbb{C}^{m\times n}, \qquad t \in [0,T], \qquad A(0) = A_0, \quad A'(0) = B_0, \tag{4.1}$$

where we assume the right-hand side F to be Lipschitz continuous with a moderate Lipschitz constant. For such equations, numerous integrators are provided in the literature. Many of them are tailored to properties of the right-hand side F or of the exact solution A of (4.1). However, solving (4.1) under a rank constraint on the approximations to the exact solution and its derivative has not yet been presented.

In this chapter, we derive dynamical low-rank integrators for second-order matrix differential equations of form (4.1) based on the projector-splitting integrator. The first scheme is constructed from the first-order formulation of the problem (4.1), which we combine with a splitting ansatz and the ideas presented in Section 3.2. Before we do so we first revise the leapfrog scheme, which presumably is the most prominent scheme for computing approximations to the solution of (4.1). One of the possibilities to derive this numerical method is based on the first-order formulation and a splitting ansatz. This is the basis for the construction of our dynamical low-rank integrator for the second-order matrix differential equation (4.1). We shall see, that the connection between the leapfrog and our low-rank scheme goes beyond the construction, and both schemes share several properties. Sections 4.2.1 and 4.2.2 are devoted to a detailed error analysis of the novel integrator. In the rest of this chapter, we design variants of the original ansatz which are tailored to specific properties of either the numerical approximation or the right-hand side of (4.1).

The construction of the schemes in Section 4.2 and Section 4.3 as well as the theoretical results in Section 4.2.1 have been submitted for publication in [Hochbruck et al., 2022a]. Here we provide more detailed information concerning the derivation of the respective dynamical low-rank integrators and the proofs in Section 4.2.1.

4.1 The leapfrog scheme

The probably most common method for solving (4.1) is the leapfrog scheme, an explicit time-integration scheme also known as the Störmer scheme in the context of astronomy or as the Verlet scheme in the context of molecular dynamics.

One possibility of deriving the leapfrog scheme is based on the first-order formulation

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} B(t) \\ F\big(A(t)\big) \end{bmatrix}, \qquad \begin{bmatrix} A(0) \\ B(0) \end{bmatrix} = \begin{bmatrix} A_0 \\ B_0 \end{bmatrix} \tag{4.2}$$

of (4.1), where $B(t) = A'(t)$. Following [Hairer et al., 2003, Section 1.5], splitting (4.2) into two subproblems yields

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} 0 \\ F\big(A(t)\big) \end{bmatrix}, \qquad \begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} B(t) \\ 0 \end{bmatrix}.$$

Both problems can be solved explicitly with solutions

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix} = \begin{bmatrix} A_0 \\ B_0 + tF(A_0) \end{bmatrix} \qquad \text{and} \qquad \begin{bmatrix} A(t) \\ B(t) \end{bmatrix} = \begin{bmatrix} A_0 + tB_0 \\ B_0 \end{bmatrix},$$

respectively. Performing a standard Strang splitting (cf. Appendix B) with step size $\tau > 0$ then yields

$$\begin{aligned} B_{k+\frac{1}{2}} &= B_k + \frac{\tau}{2} F(A_k), \\ A_{k+1} &= A_k + \tau B_{k+\frac{1}{2}}, \qquad\qquad k = 0, 1, \ldots. \\ B_{k+1} &= B_{k+\frac{1}{2}} + \frac{\tau}{2} F(A_{k+1}), \end{aligned} \tag{4.3}$$

Combining the second substep for B in the kth step with the first substep of the $(k+1)$st step leads to a variant of the leapfrog scheme, which computes approximations to A' on a *staggered* grid,

$$\begin{aligned} B_{k+\frac{1}{2}} &= B_{k-\frac{1}{2}} + \tau F(A_k), \\ A_{k+1} &= A_k + \tau B_{k+\frac{1}{2}}, \end{aligned} \qquad k = 1, 2, \ldots, \tag{4.4a}$$

where for $k = 0$ we set

$$\begin{aligned} B_{\frac{1}{2}} &= B_0 + \frac{\tau}{2} F(A_0), \\ A_1 &= A_0 + \tau B_{\frac{1}{2}}. \end{aligned} \tag{4.4b}$$

Based on the recursion (4.4), a second variant of the leapfrog scheme can be derived. For $k \geq 1$ it holds

$$\begin{aligned} A_{k+1} &= A_k + \tau B_{k+\frac{1}{2}} \\ &= A_k + \tau B_{k-\frac{1}{2}} + \tau^2 F(A_k) \\ &= A_k + (A_k - A_{k-1}) + \tau^2 F(A_k), \end{aligned}$$

or equivalently

$$A_{k+1} - 2A_k + A_{k-1} = \tau^2 F(A_k), \tag{4.5a}$$

which is also known as the *two-step* formulation of the leapfrog scheme. For the starting value

$$A_1 = A_0 + \tau B_0 + \frac{\tau^2}{2} F(A_0) \tag{4.5b}$$

it is equivalent to the one-step formulation.

Both the one-step formulation (4.4) and the two-step formulation (4.5) require the same computational effort. However, the one-step formulation is numerically more stable than (4.5), see [Hairer et al., 1993, p. 472].

The leapfrog scheme has several favorable properties. It is easy to implement, it is second-order convergent in time for nonstiff problems, and it is a symplectic integrator, which makes it of interest also in geometric numerical integration, cf. [Hairer et al., 2003, 2006]. However, the scheme does not preserve the rank of its initial values A_0, B_0, i.e., in general

$$\operatorname{rank} A_k \neq \operatorname{rank} A_0, \qquad \operatorname{rank} B_k \neq \operatorname{rank} B_0, \qquad k = 1, 2, \dots .$$

Therefore the leapfrog scheme does not qualify as a low-rank integrator for computing low-rank approximations to the exact solution to (4.1) starting from low-rank initial values $\mathbf{A}_0 \approx A_0$, $\mathbf{B}_0 \approx B_0$. Still, due to its favorable properties, we will use the leapfrog scheme as basis for the construction of a dynamical low-rank integrator tailored to the numerical integration of (4.1).

4.2 The St-LO scheme

In the spirit of Section 3.2, it appears natural to derive a dynamical low-rank integrator for second-order matrix differential equations based on the first-order formulation (4.2) of (4.1). Numerical tests indicate however, that the naive approach of applying the ideas of the first-order case directly on (4.2) yields bad results in some cases, see Section 7.2.1. We suspect that the direct application of the projector-splitting integrator for first-order problems ignores some inherent structure of the second-order problem. To overcome this, we derive a dynamical low-rank integrator based on (4.4).

The main idea is to regard the update steps for the approximations $B_{k+\frac{1}{2}} \approx A'(t_{k+\frac{1}{2}})$ and $A_{k+1} \approx A(t_{k+1})$ as displayed in (4.4) as solutions at $\sigma = \tau$ to first-order differential equations with constant right-hand sides,

$$\begin{aligned} B'(\sigma) &= F(A_k), \qquad & B(0) &= B_{k-\frac{1}{2}}, \\ A'(\sigma) &= B_{k+\frac{1}{2}}, \qquad & A(0) &= A_k. \end{aligned}$$

In order to derive a dynamical low-rank scheme, assume we are given low-rank approximations $\mathbf{A}_k \approx A(t_k)$ and $\mathbf{B}_{k-\frac{1}{2}} \approx B(t_{k-\frac{1}{2}})$. Then, low-rank approximations $\mathbf{A}_{k+1} \approx A(t_{k+1})$ and $\mathbf{B}_{k+\frac{1}{2}} \approx A'(t_{k+\frac{1}{2}})$ are obtained by approximately solving

$$\widetilde{B}'_{k-\frac{1}{2}}(\sigma) = F(\mathbf{A}_k), \qquad \widetilde{B}_{k-\frac{1}{2}}(0) = \mathbf{B}_{k-\frac{1}{2}}, \qquad \sigma \in [0, \tau], \qquad k \geq 1, \tag{4.6a}$$

$$\widetilde{A}'_k(\sigma) = \mathbf{B}_{k+\frac{1}{2}}, \qquad \widetilde{A}_k(0) = \mathbf{A}_k, \qquad \sigma \in [0, \tau], \qquad k \geq 0, \tag{4.6b}$$

employing the projector-splitting integrator. For $k = 0$, we start from

$$\widetilde{B}_0'(\sigma) = F(\mathbf{A}_0), \qquad \widetilde{B}_0(0) = \mathbf{B}_0, \qquad \sigma \in [0, \tfrac{\tau}{2}]. \tag{4.6c}$$

All problems (4.6) can be solved exactly, with solutions

$$\begin{aligned} \widetilde{B}_{k-\frac{1}{2}}(\sigma) &= \mathbf{B}_{k-\frac{1}{2}} + \sigma F(\mathbf{A}_k), & \sigma \in [0,\tau], \quad & k \geq 1, & (4.7a)\\ \widetilde{A}_k(\sigma) &= \mathbf{A}_k + \sigma \mathbf{B}_{k+\frac{1}{2}}, & \sigma \in [0,\tau], \quad & k \geq 0, & (4.7b)\\ \widetilde{B}_0(\sigma) &= \mathbf{B}_0 + \sigma F(\mathbf{A}_0), & \sigma \in [0,\tfrac{\tau}{2}]. & & (4.7c) \end{aligned}$$

Since we aim for computing dynamical low-rank approximations to given, time-dependent matrices, we can make use of the projector-splitting integrator with increments, cf. Algorithm 2.

Remark 4.1. Denote the numerical flow of the projector-splitting integrator by $\phi^{[\mathtt{prsi}]}$. For a given, time-dependent matrix $A(t)$, the computation of the low-rank approximation $\mathbf{A}_1 \approx A(\tau)$ via the projector-splitting integrator, started from the low-rank initial value $\mathbf{A}_0 \approx A(0)$ and the increment $\Delta A = \widetilde{A}_1 - \mathbf{A}_0$ is expressed as

$$\mathbf{A}_1 = \phi^{[\mathtt{prsi}]}_{\Delta A}(\widetilde{A}_1).$$

Here, $\widetilde{A}_1$ is an approximation to $A(\tau)$ of arbitrary rank. If $\widetilde{A}_1$ itself originates from a single step of some numerical scheme with flow ϕ and initial value $\mathbf{A}_0$, i.e., $\widetilde{A}_1 = \phi_\tau(\mathbf{A}_0)$, we use the notation

$$\mathbf{A}_1 = \phi^{[\mathtt{prsi}]}_{\tau} \circ \phi_\tau(\mathbf{A}_0). \qquad \diamond$$

With the above notation, the computation of the low-rank approximations can be expressed as

$$\begin{aligned} \mathbf{B}_{k+\frac{1}{2}} &= \phi^{[\mathtt{prsi}]}_{\Delta B_{k-\frac{1}{2}}}\big(\widetilde{B}_{k-\frac{1}{2}}(\tau)\big), & k \geq 1, & \quad (4.8a)\\ \mathbf{A}_{k+1} &= \phi^{[\mathtt{prsi}]}_{\Delta A_k}\big(\widetilde{A}_k(\tau)\big), & k \geq 0, & \quad (4.8b)\\ \mathbf{B}_{\frac{1}{2}} &= \phi^{[\mathtt{prsi}]}_{\Delta B_0}\big(\widetilde{B}_0(\tfrac{\tau}{2})\big). & & \quad (4.8c) \end{aligned}$$

The increments ΔA and ΔB, which are required to perform the scheme, are given as

$$\begin{aligned} \Delta B_{k-\frac{1}{2}} &= \widetilde{B}_{k-\frac{1}{2}}(\tau) - \widetilde{B}_{k-\frac{1}{2}}(0) = \tau F(\mathbf{A}_k), & k \geq 1,\\ \Delta A_k &= \widetilde{A}_k(\tau) - \widetilde{A}_k(0) = \tau \mathbf{B}_{k+\frac{1}{2}}, & k \geq 0,\\ \Delta B_0 &= \widetilde{B}_0(\tfrac{\tau}{2}) - \widetilde{B}_0(0) = \tfrac{\tau}{2} F(\mathbf{A}_0). \end{aligned}$$

The resulting dynamical low-rank integrator for second-order matrix differential equations is named *St-LO* scheme, for **St***rang splitting combined with the projector-splitting integrator as introduced by* **L***ubich and* **O***seledets*. A single time step of the scheme is presented in Algorithm 7. For decompositions

$$\mathbf{A}_k = \mathbf{U}_k \mathbf{S}_k \mathbf{V}_k^H \in \mathcal{M}_{r_\mathbf{A}}, \qquad \mathbf{B}_{k-\frac{1}{2}} = \mathbf{T}_{k-\frac{1}{2}} \mathbf{R}_{k-\frac{1}{2}} \mathbf{W}_{k-\frac{1}{2}}^H \in \mathcal{M}_{r_\mathbf{B}},$$

where

$$\begin{aligned} &\mathbf{U}_k \in \mathcal{V}_{m,r_\mathbf{A}}, \qquad \mathbf{V}_k \in \mathcal{V}_{n,r_\mathbf{A}}, \qquad \mathbf{S}_k \in \mathbb{C}^{r_\mathbf{A} \times r_\mathbf{A}}, & (4.9a)\\ &\mathbf{T}_{k-\frac{1}{2}} \in \mathcal{V}_{m,r_\mathbf{B}}, \qquad \mathbf{W}_{k-\frac{1}{2}} \in \mathcal{V}_{n,r_\mathbf{B}}, \qquad \mathbf{R}_{k-\frac{1}{2}} \in \mathbb{C}^{r_\mathbf{B} \times r_\mathbf{B}}, & (4.9b) \end{aligned}$$

this algorithm computes the factors of the approximations $\mathbf{A}_{k+1} = \mathbf{U}_{k+1} \mathbf{S}_{k+1} \mathbf{V}_{k+1}^H \approx A(t_{k+1})$ and $\mathbf{B}_{k+\frac{1}{2}} = \mathbf{T}_{k+\frac{1}{2}} \mathbf{R}_{k+\frac{1}{2}} \mathbf{W}_{k+\frac{1}{2}}^H \approx B(t_{k+\frac{1}{2}})$. These factors satisfy again (4.9).

Algorithm 7: Dynamical low-rank integrator for second-order ODEs, St-LO, single time step

```
1 stlo($\mathbf{U},\mathbf{S},\mathbf{V},\mathbf{T},\mathbf{R},\mathbf{W},r_{\mathbf{A}},r_{\mathbf{B}},\tau,F$)
  Input  : factors $\mathbf{U},\mathbf{S},\mathbf{V}$ of rank-$r_{\mathbf{A}}$ approximation $\mathbf{A}=\mathbf{U}\mathbf{S}\mathbf{V}^H \approx A(t)$ satisfying (4.9a), factors
           $\mathbf{T},\mathbf{R},\mathbf{W}$ of rank-$r_{\mathbf{B}}$ approximation $\mathbf{B}=\mathbf{T}\mathbf{R}\mathbf{W}^H \approx A'(t-\frac{\tau}{2})$ satisfying (4.9b), step
           size $\tau$, right-hand side $F$
2 B-step: $\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1,\mathbf{L} = \mathtt{prsi}\big(\mathbf{T},\mathbf{R},\mathbf{W},r_{\mathbf{B}},\Delta B\big)$    where $\Delta B = \tau F(\mathbf{U}\mathbf{S}\mathbf{V}^H)$
3 A-step: $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1,\mathbf{L}_1 = \mathtt{prsi}\big(\mathbf{U},\mathbf{S},\mathbf{V},r_{\mathbf{A}},\Delta A\big)$    where $\Delta A = \tau \mathbf{T}_1\mathbf{L}^H$
4 Return $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1,\mathbf{L}_1,\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1$
  Output: factors $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1$ of rank-$r_{\mathbf{A}}$ approximation $\mathbf{A}_1=\mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H \approx A(t+\tau)$ satisfying
          (4.9a), $\mathbf{L}_1=\mathbf{V}_1\mathbf{S}_1^H$, factors $\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1$ of rank-$r_{\mathbf{B}}$ approximation
          $\mathbf{B}_1=\mathbf{T}_1\mathbf{R}_1\mathbf{W}_1^H \approx A'(t+\frac{\tau}{2})$ satisfying (4.9b)
```

Remark 4.2. In the $\mathbf{B}$-step of the St-LO scheme, the increment ΔB is given in terms of the right-hand side F evaluated at some low-rank matrix decomposed into low-rank factors $\mathbf{U},\mathbf{S}$, and $\mathbf{V}$. The efficient implementation of this scheme requires to compute products of form $F(\mathbf{U}\mathbf{S}\mathbf{V}^H)\mathbf{W}$ and $F(\mathbf{U}\mathbf{S}\mathbf{V}^H)^H\mathbf{T}$, respectively, without computing the full-size matrix $\mathbf{U}\mathbf{S}\mathbf{V}^H$ explicitly. This is similar to the general projector-splitting integrator, cf. Algorithm 3, where the products $F(\mathbf{U}\mathbf{S}\mathbf{V}^H)\mathbf{V}$ and $F(\mathbf{U}\mathbf{S}\mathbf{V}^H)^H\mathbf{U}$ arise. For the $\mathbf{A}$ step however, due to the factorization of the increment like $\Delta A = \mathbf{T}\mathbf{L}^H$, we simply compute the products $\Delta A\mathbf{V}$ and $\Delta A^H\mathbf{U}$ by clever ordering of the matrix products. ◇

By construction, the St-LO method can be viewed as a low-rank pendant to the leapfrog scheme. For consistency, it would be favorable if the methods would yield the same approximations if the chosen ranks $r_{\mathbf{A}}$ and $r_{\mathbf{B}}$ are sufficiently large. For $r_{\mathbf{A}} = r_{\mathbf{B}} = \min\{m,n\}$, the St-LO and the leapfrog scheme indeed compute the same approximations.

Theorem 4.3. *If the approximation ranks $r_{\mathbf{A}}$ and $r_{\mathbf{B}}$ for computing low-rank approximations $\mathbf{A} \approx A$ and $\mathbf{B} \approx A'$, respectively, are chosen as $r_{\mathbf{A}} = r_{\mathbf{B}} = \min\{m,n\}$, then the solutions computed by the St-LO scheme, cf. Algorithm 7, and the leapfrog scheme (4.4) coincide. Note that this especially means $\mathbf{B}_0 = B(0)$ and $\mathbf{A}_0 = A(0)$.*

Proof. Without loss of generality, we assume $m \geq n$. Otherwise, we consider the conjugate transpose. We also introduce the set

$$\mathcal{M}_{\leq r} = \big\{ X \in \mathbb{C}^{m\times n} \;\big|\; \operatorname{rank} X \leq r \big\}$$

of complex $m \times n$ matrices with rank at most r. Obviously, any element of $\mathcal{M}_{\leq r}$ admits a factorization similarly to (2.5). However, the $\mathbf{S}$-factor is now allowed to be singular.

We start with the first substep of the integration by the St-LO scheme. We denote by $\mathbf{B}_{\frac{1}{2}} = \mathbf{T}_{\frac{1}{2}}\mathbf{R}_{\frac{1}{2}}\mathbf{W}_{\frac{1}{2}}^H \approx A'(\frac{\tau}{2})$ the low-rank approximation computed from the initial values $\mathbf{A}_0 = \mathbf{U}_0\mathbf{S}_0\mathbf{V}_0^H \in \mathcal{M}_{\leq n}$ and $\mathbf{B}_0 = \mathbf{T}_0\mathbf{R}_0\mathbf{W}_0^H \in \mathcal{M}_{\leq n}$ and let furthermore $\Delta B = \frac{\tau}{2}F(\mathbf{A}_0)$. Then it holds

$$\mathbf{V}_0, \mathbf{W}_0 \in \mathcal{V}_{n,n} \quad\iff\quad \mathbf{V}_0\mathbf{V}_0^H = \mathbf{W}_0\mathbf{W}_0^H = I_n. \tag{4.10}$$

By the update steps (3.10) of the projector-splitting integrator, $\mathbf{B}_{\frac{1}{2}}$ satisfies

$$\begin{aligned}
\mathbf{B}_{\frac{1}{2}} &= \mathbf{T}_{\frac{1}{2}}\mathbf{R}_{\frac{1}{2}}\mathbf{W}_{\frac{1}{2}}^H \\
&= \mathbf{T}_{\frac{1}{2}}(\widetilde{\mathbf{R}}_0\mathbf{W}_0^H + \mathbf{T}_{\frac{1}{2}}^H\Delta B) \\
&= \mathbf{T}_{\frac{1}{2}}\widetilde{\mathbf{R}}_0\mathbf{W}_0^H + \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B \\
&= \mathbf{T}_{\frac{1}{2}}(\widehat{\mathbf{R}}_{\frac{1}{2}} - \mathbf{T}_{\frac{1}{2}}^H\Delta B\mathbf{W}_0)\mathbf{W}_0^H + \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B \\
&= \mathbf{T}_{\frac{1}{2}}\widehat{\mathbf{R}}_{\frac{1}{2}}\mathbf{W}_0^H - \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B\Pi_{\mathbf{W}_0} + \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B \\
&= (\mathbf{T}_0\mathbf{R}_0 + \Delta B\mathbf{W}_0)\mathbf{W}_0^H - \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B\Pi_{\mathbf{W}_0} + \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B \\
&= \mathbf{B}_0 + \Delta B\Pi_{\mathbf{W}_0} - \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B\Pi_{\mathbf{W}_0} + \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B
\end{aligned}$$

in terms of the projections onto the column spaces of $\mathbf{T}_{\frac{1}{2}}$ and $\mathbf{W}_0$, respectively, cf. (2.20). Due to (4.10), $\Pi_{\mathbf{W}_0} = \mathbf{W}_0\mathbf{W}_0^H = I_n$, so that

$$\begin{aligned}
\mathbf{B}_{\frac{1}{2}} &= \mathbf{B}_0 + \Delta B - \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B + \Pi_{\mathbf{T}_{\frac{1}{2}}}\Delta B \\
&= \mathbf{B}_0 + \frac{\tau}{2}F(\mathbf{A}_0).
\end{aligned}$$

Since $\mathbf{B}_0 = B(0)$ and $\mathbf{A}_0 = A(0)$, this is indeed the first substep of the leapfrog scheme (4.4), thus $\mathbf{B}_{\frac{1}{2}} = B_{\frac{1}{2}}$. Denoting $\Delta A = \tau\mathbf{B}_{\frac{1}{2}}$, performing similar steps as above yields

$$\mathbf{A}_1 = \mathbf{A}_0 + \Delta A\Pi_{\mathbf{V}_0} - \Pi_{\mathbf{U}_1}\Delta A\Pi_{\mathbf{V}_0} + \Pi_{\mathbf{U}_1}\Delta A,$$

where $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H$ is the result of the second substep of the St-LO scheme. Again, due to (4.10) $\Pi_{\mathbf{V}_0} = I_n$ holds and hence

$$\mathbf{A}_1 = \mathbf{A}_0 + \Delta A = \mathbf{A}_0 + \tau\mathbf{B}_{\frac{1}{2}} = A(0) + \tau B_{\frac{1}{2}},$$

which coincides with the second substep of the leapfrog scheme (4.4) and thus $\mathbf{A}_1 = A_1$. A simple induction argument yields the assertion. □

Based on the one-step formulation (4.3) of the leapfrog scheme, it is possible to derive a variant of the St-LO scheme which computes approximations to A and A' on the same time-grid. However, the computational effort roughly doubles, since the right-hand side F has to be evaluated twice as many times per time step. For later use, it is convenient to formulate this variant for the modified second-order matrix differential equation

$$A''(t) = \omega^2 F\big(A(t)\big), \quad t \in [0,T], \qquad A(0) = A_0, \quad A'(0) = \omega^2 B_0, \tag{4.11}$$

for some real number $\omega \geq 0$. The equivalent first-order system reads

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} \omega^2 B(t) \\ F\big(A(t)\big) \end{bmatrix} = \begin{bmatrix} \omega^2 B(t) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ F\big(A(t)\big) \end{bmatrix}. \tag{4.12}$$

Similar steps as above lead to Algorithm 8, which is a dynamical low-rank integrator for (4.11).

Algorithm 8: Dynamical low-rank integrator for second-order ODEs (4.11), single time step

```
1 stlovar($\mathbf{U},\mathbf{S},\mathbf{V},\mathbf{T},\mathbf{R},\mathbf{W},r_{\mathbf{A}},r_{\mathbf{B}},\tau,F,\omega$)
  Input : factors $\mathbf{U},\mathbf{S},\mathbf{V}$ of rank-$r_{\mathbf{A}}$ approximation $\mathbf{A}=\mathbf{U}\mathbf{S}\mathbf{V}^H\approx A(t)$ satisfying (4.9a), factors
          $\mathbf{T},\mathbf{R},\mathbf{W}$ of rank-$r_{\mathbf{B}}$ approximation $\mathbf{B}=\mathbf{T}\mathbf{R}\mathbf{W}^H\approx A'(t)$ satisfying (4.9b), step size $\tau$,
          right-hand side $F$, weight $\omega$
2 B-step: $\mathbf{T}_{\frac12},\mathbf{R}_{\frac12},\mathbf{W}_{\frac12},\mathbf{L}=\texttt{prsi}\big(\mathbf{T},\mathbf{R},\mathbf{W},r_{\mathbf{B}},\Delta B\big)$    where $\Delta B=\frac{\tau}{2}F(\mathbf{U}\mathbf{S}\mathbf{V}^H)$
3 A-step: $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1,\mathbf{L}_1=\texttt{prsi}\big(\mathbf{U},\mathbf{S},\mathbf{V},r_{\mathbf{A}},\Delta A\big)$    where $\Delta A=\omega^2\tau\mathbf{T}_{\frac12}\mathbf{L}^H$
4 B-step: $\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1,\mathbf{L}=\texttt{prsi}\big(\mathbf{T}_{\frac12},\mathbf{R}_{\frac12},\mathbf{W}_{\frac12},r_{\mathbf{B}},\Delta B\big)$    where $\Delta B=\frac{\tau}{2}F(\mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H)$
5 Return $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1,\mathbf{L}_1,\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1$
  Output: factors $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1$ of rank-$r_{\mathbf{A}}$ approximation $\mathbf{A}_1=\mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H\approx A(t+\tau)$ satisfying
          (4.9a), $\mathbf{L}_1=\mathbf{V}_1\mathbf{S}_1^H$, factors $\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1$ of rank-$r_{\mathbf{B}}$ approximation
          $\mathbf{B}_1=\mathbf{T}_1\mathbf{R}_1\mathbf{W}_1^H\approx A'(t+\tau)$ satisfying (4.9b)
```

Lastly, we derive a dynamical low-rank integrator for second-order matrix differential equations (4.1) based on the two-step formulation (4.5) of the leapfrog scheme,

$$\begin{aligned} A_{k+1}-2A_k+A_{k-1}&=\tau^2F(A_k), \qquad k\geq 1,\\ A_1&=A_0+\tau B_0+\frac{\tau^2}{2}F(A_0). \end{aligned}$$

We name the new method `stlotwostep` scheme, although it is not dircetly derived from a Strang splitting.

Given rank-$r_{\mathbf{A}}$ and rank-$r_{\mathbf{B}}$ approximations $\mathbf{A}_0\approx A_0$, $\mathbf{B}_0\approx B_0$, we compute $\mathbf{A}_1\approx A(t_1)$ by approximating the starting value A_1 of the leapfrog scheme employing the projector-splitting integrator, cf. Algorithm 2. I.e., starting from $\mathbf{A}_0$ we perform one step of the `prsi` method with increment $\Delta A=\widetilde{A}_1-\mathbf{A}_0$, where

$$\widetilde{A}_1=\mathbf{A}_0+\tau\mathbf{B}_0+\frac{\tau^2}{2}F(\mathbf{A}_0).$$

Similarly, for $k\geq 1$, we obtain the low-rank approximations $\mathbf{A}_{k+1}\approx A(t_{k+1})$ via

$$\widetilde{A}_{k+1}=2\mathbf{A}_k-\mathbf{A}_{k-1}+\tau^2F(\mathbf{A}_k), \tag{4.13a}$$

$$\mathbf{A}_{k+1}=\phi^{[\texttt{prsi}]}_{\Delta A}(\mathbf{A}_k), \qquad \Delta A=\widetilde{A}_{k+1}-\mathbf{A}_k. \tag{4.13b}$$

Here we made use of the notation introduced in Remark 4.1 for the flow of the `prsi` scheme. Obviously, the matrix $\widetilde{A}_{k+1}$ in (4.13) should not be computed explicitly. For an efficient scheme, its representation in terms of the low-rank matrices $\mathbf{A}_k$ and $\mathbf{A}_{k-1}$ (or more precisely: in terms of their low-rank factors) should be incorporated in the implementation.

Remark 4.4. The update steps for computing $\mathbf{A}_1=\mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H\approx A(t_1)$ by the projector-splitting integrator are given as

$$\begin{aligned} \mathbf{U}_1\widehat{\mathbf{S}}_1&=\mathbf{U}_0\mathbf{S}_0+\Delta A\mathbf{V}_0,\\ \widetilde{\mathbf{S}}_0&=\widehat{\mathbf{S}}_1-\mathbf{U}_1^H\Delta A\mathbf{V}_0,\\ \mathbf{V}_1\mathbf{S}_1^H&=\mathbf{V}_0\widetilde{\mathbf{S}}_0^H+\Delta A^H\mathbf{U}_1, \end{aligned}$$

cf. (3.10). If the projector-splitting integrator is started from the low-rank matrix $\mathbf{A}_0 = \mathbf{U}_0\mathbf{S}_0\mathbf{V}_0^H$, the increment takes the form

$$\Delta A = \widetilde{A}_1 - \mathbf{A}_0 = \widetilde{A}_1 - \mathbf{U}_0\mathbf{S}_0\mathbf{V}_0^H.$$

Then it is

$$\begin{aligned}\mathbf{U}_1\widehat{\mathbf{S}}_1 &= \mathbf{U}_0\mathbf{S}_0 + \Delta A\mathbf{V}_0\\ &= \mathbf{U}_0\mathbf{S}_0 + (\widetilde{A}_1 - \mathbf{A}_0)\mathbf{V}_0\\ &= \mathbf{U}_0\mathbf{S}_0 + \widetilde{A}_1\mathbf{V}_0 - \mathbf{U}_0\mathbf{S}_0\mathbf{V}_0^H\mathbf{V}_0\\ &= \widetilde{A}_1\mathbf{V}_0.\end{aligned}$$

This yields

$$\widetilde{\mathbf{S}}_0 = \widehat{\mathbf{S}}_1 - \mathbf{U}_1^H\left(\mathbf{U}_1\widehat{\mathbf{S}}_1 - \mathbf{U}_0\mathbf{S}_0\right) = \mathbf{U}_1^H\mathbf{U}_0\mathbf{S}_0$$

and

$$\begin{aligned}\mathbf{V}_1\mathbf{S}_1^H &= \mathbf{V}_0\widetilde{\mathbf{S}}_0^H + \Delta A^H\mathbf{U}_1\\ &= \mathbf{V}_0\mathbf{S}_0^H\mathbf{U}_0^H\mathbf{U}_1 + (\widetilde{A}_1 - \mathbf{A}_0)^H\mathbf{U}_1\\ &= \mathbf{V}_0\mathbf{S}_0^H\mathbf{U}_0^H\mathbf{U}_1 + \widetilde{A}_1^H\mathbf{U}_1 - \mathbf{V}_0\mathbf{S}_0^H\mathbf{U}_0^H\mathbf{U}_1\\ &= \widetilde{A}_1^H\mathbf{U}_1.\end{aligned}$$

We thus end up with only two QR decompositions

$$\begin{aligned}\mathbf{U}_1\widehat{\mathbf{S}}_1 &= \widetilde{A}_1\mathbf{V}_0,\\ \mathbf{V}_1\mathbf{S}_1^H &= \widetilde{A}_1^H\mathbf{U}_1,\end{aligned}\tag{4.14}$$

for computing $\mathbf{A}_1 \approx \widetilde{A}_1 \approx A(t_1)$. $\diamond$

4.2.1 Error analysis of the St-LO scheme

In the following, we analyze the error of the St-LO scheme (Algorithm 7) when applied to (4.1), where the right-hand side F is assumed to be Lipschitz-continuous with a moderate Lipschitz constant L_F, i.e.,

$$\|F(Y) - F(\widetilde{Y})\| \le \mathrm{L}_F\|Y - \widetilde{Y}\| \qquad \text{for all } Y, \widetilde{Y} \in \mathbb{C}^{m\times n}. \tag{4.15}$$

Assumption 4.5. *The exact solution $A : [0,T] \to \mathbb{C}^{m\times n}$ of (4.1) is in $\mathcal{C}^4([0,T])$. Furthermore, there are low-rank matrices $\mathbf{X}_A(t) \in \mathcal{M}_{r_\mathbf{A}}, \mathbf{X}_B(t) \in \mathcal{M}_{r_\mathbf{B}}$ such that*

$$A(t) = \mathbf{X}_A(t) + R_A(t), \qquad \|R_A(0)\| \le \varrho_A, \qquad \|R_A'(t)\| \le \varrho_A', \tag{4.16a}$$

$$B(t) = A'(t) = \mathbf{X}_B(t) + R_B(t), \qquad \|R_B(0)\| \le \varrho_B, \qquad \|R_B'(t)\| \le \varrho_B'. \tag{4.16b}$$

Additionally, there exist sufficiently large constants γ_A and γ_B, such that for all $Y_A(t), Y_B(t) \in \mathbb{C}^{m\times n}$ with

$$\|A(t) - Y_A(t)\| \le \gamma_A, \qquad \|B(t) - Y_B(t)\| \le \gamma_B,$$

the bounds (4.16) are also satisfied, i.e., there are low-rank matrices $\widetilde{\mathbf{X}}_A(t) \in \mathcal{M}_{r_\mathbf{A}}, \widetilde{\mathbf{X}}_B(t) \in \mathcal{M}_{r_\mathbf{B}}$ such that

$$\begin{aligned} Y_A(t) &= \widetilde{\mathbf{X}}_A(t) + \widetilde{R}_A(t), \qquad \|\widetilde{R}_A(0)\| \le \varrho_A, \qquad \|\widetilde{R}'_A(t)\| \le \varrho'_A, \\ Y_B(t) &= \widetilde{\mathbf{X}}_B(t) + \widetilde{R}_B(t), \qquad \|\widetilde{R}_B(0)\| \le \varrho_B, \qquad \|\widetilde{R}'_B(t)\| \le \varrho'_B. \end{aligned}$$

Recall, that we denote the low-rank approximations defined in (4.8) by $\mathbf{A}_k \approx A(t_k)$ and $\mathbf{B}_{k+\frac{1}{2}} \approx B(t_{k+\frac{1}{2}})$, respectively. Further we have for $\sigma \in [0, \tau]$ the auxiliary quantities

$$\begin{aligned} \widetilde{A}_k(\sigma) &= \mathbf{A}_k + \sigma \mathbf{B}_{k+\frac{1}{2}}, \\ \widetilde{B}_{k-\frac{1}{2}}(\sigma) &= \mathbf{B}_{k-\frac{1}{2}} + \sigma F(\mathbf{A}_k), \end{aligned}$$

cf. (4.7).

For better readability, we introduce the following notation: For approximations $\mathbf{A}_k \approx A(t_k)$, $\mathbf{B}_{k+\frac{1}{2}} \approx B(t_{k+\frac{1}{2}})$ obtained by the St-LO scheme and $\widetilde{A}_k$ and $\widetilde{B}_{k-\frac{1}{2}}$, we write

$$\begin{aligned} E_A^k &= \|A(t_k) - \mathbf{A}_k\|, & E_B^{k+\frac{1}{2}} &= \|B(t_{k+\frac{1}{2}}) - \mathbf{B}_{k+\frac{1}{2}}\|, && k \ge 0, \\ \widetilde{E}_A^0 &= 0, & \widetilde{E}_B^{\frac{1}{2}} &= \|B(t_{\frac{1}{2}}) - \widetilde{B}_0(t_{\frac{1}{2}})\|, \\ \widetilde{E}_A^k &= \|A(t_k) - \widetilde{A}_{k-1}(\tau)\|, & \widetilde{E}_B^{k+\frac{1}{2}} &= \|B(t_{k+\frac{1}{2}}) - \widetilde{B}_{k-\frac{1}{2}}(\tau)\|, && k \ge 1, \\ \widehat{E}_A^0 &= E_A^0, & \widehat{E}_B^{\frac{1}{2}} &= \|\widetilde{B}_0(t_{\frac{1}{2}}) - \mathbf{B}_{\frac{1}{2}}\|, \\ \widehat{E}_A^k &= \|\widetilde{A}_{k-1}(\tau) - \mathbf{A}_k\|, & \widehat{E}_B^{k+\frac{1}{2}} &= \|\widetilde{B}_{k-\frac{1}{2}}(\tau) - \mathbf{B}_{k+\frac{1}{2}}\|, && k \ge 1. \end{aligned}$$

By the triangle inequality, we have

$$E_B^{k+\frac{1}{2}} \le \widetilde{E}_B^{k+\frac{1}{2}} + \widehat{E}_B^{k+\frac{1}{2}} \qquad \text{and} \qquad E_A^{k+1} \le \widetilde{E}_A^{k+1} + \widehat{E}_A^{k+1}, \qquad k \ge 0. \tag{4.19}$$

The analysis of the St-LO scheme is performed in two steps. First, we derive recursive inequalities for E_A^{k+1} and $E_B^{k+\frac{1}{2}}$. Essential in their derivation is the Taylor series expansion of the exact solution A and its derivative. As the St-LO scheme yields approximations to A' on a staggered grid, we need an expansion of the exact solution where the derivative is evaluated on the staggered grid as well. For this, we recall Taylor's theorem:

Theorem 4.6 (Taylor's theorem)**.** *Let $u : I \subseteq \mathbb{R} \to \mathbb{R}^d, t \mapsto u(t)$ be $(p+1)$ times differentiable. Then it holds for all a and t in I*

$$u(t) = \sum_{j=0}^{p} \frac{u^{(j)}(a)}{j!}(t-a)^j + \int_a^t \frac{(t-\xi)^p}{p!} u^{(p+1)}(\xi) \, \mathrm{d}\xi.$$

The remainder satisfies the bound

$$\left\| \int_a^t \frac{(t-\xi)^p}{p!} u^{(p+1)}(\xi) \, \mathrm{d}\xi \right\| \le \max_{\xi \in [a,t]} \|u^{(p+1)}(\xi)\| \frac{|(t-a)^{p+1}|}{(p+1)!}.$$

Corollary 4.7. *Let $u : [0, T] \to \mathbb{C}^{m \times n}$ with $u \in \mathcal{C}^3([0, T])$ and $\tau > 0$. Then, as long as $t + \tau \le T$,*

$$u(t+\tau) = u(t) + \tau u'\big(t + \frac{\tau}{2}\big) + \Delta_u,$$

where the remainder Δ_u is bounded by

$$\|\Delta_u\| \le \frac{\tau^3}{24} \max_{t \in [0,T]} \|u'''(t)\|.$$

Proof. By Theorem 4.6 it is

$$u(t+\tau) = u\big(t+\tfrac{\tau}{2}\big) + \frac{\tau}{2}u'\big(t+\tfrac{\tau}{2}\big) + \frac{\tau^2}{4}u''\big(t+\tfrac{\tau}{2}\big) + \int_{t+\frac{\tau}{2}}^{t+\tau} \frac{(t+\tau-\xi)^2}{2}u'''(\xi)\,\mathrm{d}\xi,$$
$$u(t) = u\big(t+\tfrac{\tau}{2}\big) - \frac{\tau}{2}u'\big(t+\tfrac{\tau}{2}\big) + \frac{\tau^2}{4}u''\big(t+\tfrac{\tau}{2}\big) - \int_{t}^{t+\frac{\tau}{2}} \frac{(t-\xi)^2}{2}u'''(\xi)\,\mathrm{d}\xi.$$

Subtracting both equations yields

$$u(t+\tau) - u(t) = \tau u'\big(t+\tfrac{\tau}{2}\big) + \Delta_u,$$

where

$$\begin{aligned}\|\Delta_u\| &= \left\| \int_t^{t+\frac{\tau}{2}} \frac{(t-\xi)^2}{2}u'''(\xi)\,\mathrm{d}\xi + \int_{t+\frac{\tau}{2}}^{t+\tau} \frac{(t+\tau-\xi)^2}{2}u'''(\xi)\,\mathrm{d}\xi \right\| \\ &\le \max_{\xi\in[t,t+\frac{\tau}{2}]} \|u'''(\xi)\| \int_t^{t+\frac{\tau}{2}} \frac{(t-\xi)^2}{2}\,\mathrm{d}\xi + \max_{\xi\in[t+\frac{\tau}{2},t+\tau]} \|u'''(\xi)\| \int_{t+\frac{\tau}{2}}^{t+\tau} \frac{(t+\tau-\xi)^2}{2}\,\mathrm{d}\xi \\ &\le \frac{\tau^3}{48}\Big(\max_{\xi\in[t,t+\frac{\tau}{2}]} \|u'''(\xi)\| + \max_{\xi\in[t+\frac{\tau}{2},t+\tau]} \|u'''(\xi)\| \Big) \\ &\le \frac{\tau^3}{24} \max_{t\in[0,T]} \|u'''(t)\|,\end{aligned}$$

and thus the assertion. □

With Corollary 4.7, we are able to derive the desired recursive inequalities:

Lemma 4.8. *Let $A : [0,T] \to \mathbb{C}^{m\times n}$ with $A \in \mathcal{C}^4([0,T])$ be the exact solution of (4.1) with initial values $A_0, B_0 \in \mathbb{C}^{m\times n}$ and $B = A'$. Moreover, denote by $\mathbf{B}_{k-\frac{1}{2}} \in \mathcal{M}_{r_\mathbf{B}}$ and $\mathbf{A}_k \in \mathcal{M}_{r_\mathbf{A}}$ the low-rank approximations obtained by the St-LO scheme after k steps started with initial values $\mathbf{A}_0 \in \mathcal{M}_{r_\mathbf{A}}, \mathbf{B}_0 \in \mathcal{M}_{r_\mathbf{B}}$. Then, the errors introduced in (4.18) satisfy*

$$E_B^{\frac{1}{2}} \le E_B^0 + \frac{\tau}{2}\mathrm{L}_F E_A^0 + \widehat{E}_B^{\frac{1}{2}} + C_B^{\mathrm{LF}}\tau^2, \tag{4.20a}$$
$$E_B^{k+\frac{1}{2}} \le E_B^{k-\frac{1}{2}} + \tau\mathrm{L}_F E_A^k + \widehat{E}_B^{k+\frac{1}{2}} + C_B^{\mathrm{LF}}\tau^3 \qquad \text{for } k \ge 1, \tag{4.20b}$$
$$E_A^{k+1} \le E_A^k + \tau E_B^{k+\frac{1}{2}} + \widehat{E}_A^{k+1} + C_A^{\mathrm{LF}}\tau^3 \qquad \text{for } k \ge 0, \tag{4.20c}$$

where C_A^{LF} and C_B^{LF} are given explicitly as

$$C_A^{\mathrm{LF}} = \max_{t\in[0,T]} \frac{1}{24}\|A'''(t)\|, \qquad C_B^{\mathrm{LF}} = \max\Big\{ \max_{t\in[0,\frac{\tau}{2}]} \frac{1}{8}\|A'''(t)\|,\ \max_{t\in[0,T]} \frac{1}{24}\|A^{(4)}(t)\| \Big\}.$$

Proof. We use Taylor series expansion for $B(t_{\frac{1}{2}})$ and get

$$\begin{aligned} B(t_{\frac{1}{2}}) &= B(0) + \frac{\tau}{2}B'(0) + \int_0^{\frac{\tau}{2}} \big(\tfrac{\tau}{2} - \xi\big)B''(\xi)\,\mathrm{d}\xi \\ &= B_0 + \frac{\tau}{2}F(A_0) + \int_0^{\frac{\tau}{2}} \big(\tfrac{\tau}{2} - \xi\big)B''(\xi)\,\mathrm{d}\xi. \end{aligned}$$

Hence

$$\|B(t_{\frac{1}{2}}) - \big(B_0 + \frac{\tau}{2}F(A_0)\big)\| \le \frac{\tau^2}{8} \max_{t\in[0,\frac{\tau}{2}]} \|A'''(t)\| \le C_B^{\mathrm{LF}}\tau^2. \tag{4.21}$$

Furthermore we have by Corollary 4.7

$$\|A(t_{k+1}) - \big(A(t_k) + \tau B(t_{k+\frac{1}{2}})\big)\| \le C_A^{\mathrm{LF}} \tau^3, \qquad k \ge 0, \tag{4.22a}$$

$$\|B(t_{k+\frac{1}{2}}) - \big(B(t_{k-\frac{1}{2}}) + \tau F(A(t_k))\big)\| \le C_B^{\mathrm{LF}} \tau^3, \qquad k \ge 1. \tag{4.22b}$$

Hence for $k = 0$, we have

$$\begin{aligned}\widetilde{E}_B^{\frac{1}{2}} &\le \|B_0 + \frac{\tau}{2} F(A_0) - (\mathbf{B}_0 + \frac{\tau}{2} F(\mathbf{A}_0))\| + C_B^{\mathrm{LF}} \tau^2 \\ &\le E_B^0 + \frac{\tau}{2} \mathrm{L}_F E_A^0 + C_B^{\mathrm{LF}} \tau^2 \end{aligned} \tag{4.23}$$

by (4.21), (4.7c), and (4.15). Employing (4.19) shows (4.20a). Likewise, for $k \ge 1$ we have by (4.7), (4.15), and (4.22b)

$$\widetilde{E}_B^{k+\frac{1}{2}} \le E_B^{k-\frac{1}{2}} + \tau \mathrm{L}_F E_A^k + C_B^{\mathrm{LF}} \tau^3, \tag{4.24}$$

and by (4.19) thus (4.20b).

For $k \ge 0$, (4.22a) yields together with (4.19) and (4.7)

$$\begin{aligned} E_A^{k+1} &\le \|A(t_k) + \tau A'(t_{k+\frac{1}{2}}) - \big(\mathbf{A}_k + \tau \mathbf{B}_{k+\frac{1}{2}}\big)\| + \widehat{E}_A^{k+1} + C_A^{\mathrm{LF}} \tau^3 \\ &\le E_A^k + \tau E_B^{k+\frac{1}{2}} + \widehat{E}_A^{k+1} + C_A^{\mathrm{LF}} \tau^3, \end{aligned}$$

hence (4.20c). □

For the next considerations, it is helpful to have a slightly different representation of the errors E_A^{k+1} and $E_B^{k+\frac{1}{2}}$.

Corollary 4.9. *Let the assumptions of* Lemma 4.8 *be satisfied. Then it holds*

$$E_A^1 \le (1 + \frac{\tau^2}{2} \mathrm{L}_F) E_A^0 + \tau E_B^0 + \tau \widehat{E}_B^{\frac{1}{2}} + \widehat{E}_A^1 + (C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}}) \tau^3, \tag{4.25a}$$

$$E_A^{k+1} \le (1 + \tau^2 \mathrm{L}_F) E_A^k + \tau E_B^{k-\frac{1}{2}} + \tau \widehat{E}_B^{k+\frac{1}{2}} + \widehat{E}_A^{k+1} + (C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}} \tau) \tau^3 \qquad \textit{for } k \ge 1, \tag{4.25b}$$

$$E_B^{k+\frac{1}{2}} \le (1 + \tau^2 \mathrm{L}_F) E_B^{k-\frac{1}{2}} + \tau \mathrm{L}_F E_A^{k-1} + \tau \mathrm{L}_F \widehat{E}_A^k + \widehat{E}_B^{k+\frac{1}{2}} + C_A^{\mathrm{LF}} \mathrm{L}_F \tau^4 + C_B^{\mathrm{LF}} \tau^3 \qquad \textit{for } k \ge 1. \tag{4.25c}$$

Proof. (4.25a) is obtained by replacing $E_B^{\frac{1}{2}}$ in (4.20c) for $k = 0$ by (4.20a). Similarly, (4.25b) follows from replacing $E_B^{k+\frac{1}{2}}$ in (4.20c) by (4.20b). Note, that (4.25a) and (4.25b) imply

$$\widetilde{E}_A^1 \le \big(1 + \frac{\tau^2}{2} \mathrm{L}_F\big) E_A^0 + \tau E_B^0 + \tau \widehat{E}_B^{\frac{1}{2}} + (C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}}) \tau^3 \tag{4.26}$$

and

$$\widetilde{E}_A^{k+1} \le (1 + \tau^2 \mathrm{L}_F) E_A^k + \tau E_B^{k-\frac{1}{2}} + \tau \widehat{E}_B^{k+\frac{1}{2}} + (C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}} \tau) \tau^3. \tag{4.27}$$

Lastly, replacing E_A^k in (4.20b) by (4.20c) for $k - 1$ yields (4.25c). □

In [Kieri et al., 2016, Section 2.6.1] it was shown that the error between a time-dependent matrix $A(t) \in \mathbb{C}^{m \times n}$ satisfying (4.16a) and the rank-$r_{\mathbf{A}}$ approximation $Y_1 \approx A(\tau)$ computed with the projector-splitting integrator (Algorithm 2) started from $X_A(0) \in \mathcal{M}_{r_{\mathbf{A}}}$ is bounded by

$$\|A(\tau) - Y_1\| \le \varrho_A + 7 \tau \varrho_A'. \tag{4.28}$$

In the next lemma we eliminate $\widehat{E}_B^{k+\frac{1}{2}}$ and $\widehat{E}_A^{k+1}$ from (4.25) by using this relation. Also, we substitute (4.20b) into (4.25b) to obtain a recursion of E_A^{k+1} in terms of the global errors E_A^j for $j = 0, \dots, k$.

Lemma 4.10. *Let the assumptions of Lemma 4.8 be satisfied. Furthermore, assume that*

$$E_A^0 \le \varrho_A, \qquad E_B^0 \le \varrho_B. \tag{4.29}$$

If Assumption 4.5 holds, then for all k such that $t_{k+3} \le T$, the errors $E_B^{k+\frac{1}{2}}$ and E_A^{k+1} defined in (4.18) satisfy

$$E_B^{k+\frac{1}{2}} \le \varrho_B + 7t_{k+1}\varrho_B' + \tau \mathrm{L}_F \sum_{j=0}^{k} E_A^j + C_B^{\mathrm{LF}}\tau^2(1+t_k) \tag{4.30a}$$

and

$$\begin{aligned} E_A^{k+1} \le \varrho_A + t_{k+1}\varrho_B + \tau \mathrm{L}_F \sum_{j=0}^{k} t_{k+1-j}E_A^j + 7t_{k+1}\varrho_A' + \frac{7}{2}t_{k+1}t_{k+2}\varrho_B' \\ + \left((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_{k+1} + \frac{1}{2}t_k t_{k+1} C_B^{\mathrm{LF}} \right)\tau^2, \end{aligned} \tag{4.30b}$$

respectively.

Proof. The proof is done by induction on k. First, we show that the errors $\widetilde{E}_A^{k+1}$ and $\widetilde{E}_B^{k+\frac{1}{2}}$ are uniformly bounded by suitable constants γ_A and γ_B. Then the auxiliary solutions $\widetilde{A}$ and $\widetilde{B}$ are sufficiently close to the exact solutions A and B, and hence they admit decompositions like (4.16) by Assumption 4.5. Since the approximations $\mathbf{A}$ and $\mathbf{B}$ are rank-$r_{\mathbf{A}}$ and rank-$r_{\mathbf{B}}$ approximations to $\widetilde{A}$ and $\widetilde{B}$ computed with the projector-splitting integrator (cf. Algorithm 2) started from initial values of rank $r_{\mathbf{A}}$ and $r_{\mathbf{B}}$, respectively, the local errors $\widehat{E}_A^{k+1}$ and $\widehat{E}_B^{k+\frac{1}{2}}$ are bounded by

$$\widehat{E}_B^{\frac{1}{2}} \le \frac{7}{2}\tau\varrho_B', \quad \widehat{E}_B^{k+\frac{1}{2}} \le 7\tau\varrho_B', \quad k \ge 1, \qquad \widehat{E}_A^{k+1} \le 7\tau\varrho_A', \quad k \ge 0, \tag{4.31}$$

cf. (4.28). The estimate on the global error then follows directly from (4.19).

For $k = 0$ we deduce from (4.23) and (4.29)

$$\widetilde{E}_B^{\frac{1}{2}} \le \varrho_B + \frac{\tau}{2}\mathrm{L}_F\varrho_A + C_B^{\mathrm{LF}}\tau^2.$$

By Assumption 4.5, if $\gamma_B \ge \varrho_B + \frac{\tau}{2}\mathrm{L}_F\varrho_A + C_B^{\mathrm{LF}}\tau^2$, then by (4.20a) and (4.31) it holds

$$\begin{aligned} E_B^{\frac{1}{2}} &\le \varrho_B + \frac{\tau}{2}\mathrm{L}_F E_A^0 + 7\frac{\tau}{2}\varrho_B' + C_B^{\mathrm{LF}}\tau^2 \\ &< \varrho_B + \tau\mathrm{L}_F E_A^0 + 7t_1\varrho_B' + C_B^{\mathrm{LF}}\tau^2, \end{aligned} \tag{4.32}$$

which precisely is (4.30a) for $k = 0$.

Likewise, by (4.26), (4.29), and (4.32) it holds

$$\widetilde{E}_A^1 \le \varrho_A + \tau^2\mathrm{L}_F E_A^0 + \tau\varrho_B + 7\tau^2\varrho_B' + (C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})\tau^3. \tag{4.33}$$

Choosing γ_A as the right-hand side of (4.33), we get by Assumption 4.5 and (4.31)

$$E_A^1 \le \varrho_A + \tau^2\mathrm{L}_F E_A^0 + t_1\varrho_B + 7t_1\varrho_A' + \frac{7}{2}t_1t_2\varrho_B' + (C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_1\tau^2,$$

which shows that (4.30b) indeed holds for $k = 0$.

Assume now that (4.30) is true for some arbitrary, but fixed $k-1 \in \mathbb{N}_0$. Using the Gronwall-type Lemma 4.13 below we find from (4.30b) for $j = 1, \ldots, k$

$$E_A^j \leq \mathrm{e}^{\sqrt{\mathrm{L}_F} t_k}\, M_A^j, \tag{4.34}$$

where

$$M_A^j = \varrho_A + t_j \varrho_B + 7 t_j \varrho_A' + \frac{7}{2} t_j t_{j+1} \varrho_B' + \big((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}}) t_j + \frac{1}{2} t_{j-1} t_j C_B^{\mathrm{LF}}\big)\tau^2.$$

By (4.29), this bound is also valid for $j = 0$. From (4.24) and (4.30a) we obtain

$$\begin{aligned}
\widetilde{E}_B^{k+\frac{1}{2}} &\leq E_B^{k-\frac{1}{2}} + \tau \mathrm{L}_F E_A^k + C_B^{\mathrm{LF}} \tau^3 \\
&\leq \Big(\varrho_B + 7 t_k \varrho_B' + \tau \mathrm{L}_F \sum_{j=0}^{k-1} E_A^j + C_B^{\mathrm{LF}} \tau^2 (1 + t_{k-1})\Big) + \tau \mathrm{L}_F E_A^k + C_B^{\mathrm{LF}} \tau^3 \\
&= \varrho_B + 7 t_k \varrho_B' + \tau \mathrm{L}_F \sum_{j=0}^{k} E_A^j + C_B^{\mathrm{LF}} \tau^2 (1 + t_k).
\end{aligned} \tag{4.35}$$

Plugging (4.34) into (4.35) yields

$$\begin{aligned}
\widetilde{E}_B^{k+\frac{1}{2}} &\leq \varrho_B + 7 t_k \varrho_B' + C_B^{\mathrm{LF}} \tau^2 (1 + t_k) \\
&\quad + \mathrm{L}_F\, \mathrm{e}^{\sqrt{\mathrm{L}_F} t_k} \sum_{j=0}^{k} \tau \Big(\varrho_A + t_j \varrho_B + 7 t_j \varrho_A' + \frac{7}{2} t_j t_{j+1} \varrho_B' + \big((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}}) t_j + \frac{1}{2} t_{j-1} t_j C_B^{\mathrm{LF}}\big)\tau^2\Big).
\end{aligned}$$

Due to

$$\sum_{j=0}^{k} \tau = t_{k+1}, \qquad \sum_{j=0}^{k} \tau t_j = \frac{1}{2} t_k t_{k+1},$$

$$\sum_{j=0}^{k} \tau t_j t_{j+1} = \frac{1}{3} t_k t_{k+1} t_{k+2}, \qquad \sum_{j=0}^{k} \tau t_{j-1} t_j = \frac{1}{3} t_{k-1} t_k t_{k+1},$$

we have

$$\begin{aligned}
\widetilde{E}_B^{k+\frac{1}{2}} &\leq \varrho_B + 7 t_k \varrho_B' + C_B^{\mathrm{LF}} \tau^2 (1 + t_k) \\
&\quad + \mathrm{L}_F\, \mathrm{e}^{\sqrt{\mathrm{L}_F} t_k} \Big(t_{k+1} \varrho_A + \frac{1}{2} t_k t_{k+1} \varrho_B + \frac{7}{2} t_k t_{k+1} \varrho_A' + \frac{7}{6} t_k t_{k+1} t_{k+2} \varrho_B' \\
&\qquad + \big(\frac{1}{2}(C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}}) t_k t_{k+1} + \frac{1}{6} C_B^{\mathrm{LF}} t_{k-1} t_k t_{k+1}\big)\tau^2\Big) \\
&= \Big(\varrho_B + 7 t_k \varrho_B' + \mathrm{L}_F\, \mathrm{e}^{\sqrt{\mathrm{L}_F} t_k} \big(t_{k+1} \varrho_A + \frac{1}{2} t_k t_{k+1} \varrho_B + \frac{7}{2} t_k t_{k+1} \varrho_A' + \frac{7}{6} t_k t_{k+1} t_{k+2} \varrho_B'\big)\Big) \\
&\quad + \tau^2 \Big(C_B^{\mathrm{LF}} (1 + t_k) + \mathrm{L}_F\, \mathrm{e}^{\sqrt{\mathrm{L}_F} t_k} \big(\frac{1}{2}(C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}}) t_k t_{k+1} + \frac{1}{6} C_B^{\mathrm{LF}} t_{k-1} t_k t_{k+1}\big)\Big).
\end{aligned}$$

Thus, for $0 \leq t_{k+2} \leq T$, there are constants $C_B(T)$, $\widetilde{C}_B(T)$ depending on $\mathrm{L}_F, \varrho_A, \varrho_B, \varrho_A', \varrho_B', C_A^{\mathrm{LF}}$, and C_B^{LF} such that

$$\widetilde{E}_B^{k+\frac{1}{2}} \leq C_B(T) + \tau^2 \widetilde{C}_B(T).$$

If Assumption 4.5 holds for some $\gamma_B \geq C_B(T) + \tau^2 \widetilde{C}_B(T)$, we obtain from (4.31) the estimate $\widehat{E}_B^{k+\frac{1}{2}} \leq 7\tau \varrho_B'$. Then (4.19) yields together with (4.35)

$$E_B^{k+\frac{1}{2}} \leq \varrho_B + 7 t_{k+1} \varrho_B' + \tau \mathrm{L}_F \sum_{j=0}^{k} E_A^j + C_B^{\mathrm{LF}} \tau^2 (1 + t_k),$$

which is (4.30a).

Similarly, using (4.27) and the induction hypothesis, we get

$$\begin{aligned}\widetilde{E}_A^{k+1} &\le \Big(\varrho_A + t_k\varrho_B + \tau \mathrm{L}_F \sum_{j=0}^{k-1} t_{k-j} E_A^j + 7t_k\varrho_A' + \frac{7}{2}t_k t_{k+1}\varrho_B' \\ &\qquad\qquad + \big((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_k + \frac{1}{2}t_{k-1}t_k C_B^{\mathrm{LF}}\big)\tau^2\Big) + \tau^2 \mathrm{L}_F E_A^k \\ &\quad + \tau\Big(\varrho_B + 7t_k\varrho_B' + \tau\mathrm{L}_F \sum_{j=0}^{k-1} E_A^j + C_B^{\mathrm{LF}}\tau^2(1+t_{k-1})\Big) \\ &\quad + 7\tau^2\varrho_B' + (C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}}\tau)\tau^3 \\ &= \varrho_A + t_{k+1}\varrho_B + \tau\mathrm{L}_F \sum_{j=0}^{k} t_{k+1-j}E_A^j + 7t_k\varrho_A' + \frac{7}{2}t_{k+1}t_{k+2}\varrho_B' \\ &\quad + \big((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_{k+1} + \frac{1}{2}t_k t_{k+1} C_B^{\mathrm{LF}}\big)\tau^2.\end{aligned} \tag{4.36}$$

Employing the bound (4.34) on E_A^j yields together with the identities

$$\sum_{j=0}^{k}\tau t_{k+1-j} = \frac{1}{2}t_{k+1}t_{k+2}, \qquad \sum_{j=0}^{k}\tau t_{k+1-j}t_j = \frac{1}{6}t_k t_{k+1}t_{k+2},$$

$$\sum_{j=0}^{k}\tau t_{k+1-j}t_j t_{j+1} = \frac{1}{12}t_k t_{k+1}t_{k+2}t_{k+3}, \qquad \sum_{j=0}^{k}\tau t_{k+1-j}t_{j-1}t_j = \frac{1}{12}t_{k-1}t_k t_{k+1}t_{k+2},$$

the bound

$$\begin{aligned}\widetilde{E}_A^{k+1} &\le \varrho_A + t_{k+1}\varrho_B + 7t_k\varrho_A' + \frac{7}{2}t_{k+1}t_{k+2}\varrho_B' + \big((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_{k+1} + \frac{1}{2}t_k t_{k+1}C_B^{\mathrm{LF}}\big)\tau^2 \\ &\quad + \mathrm{L}_F\,\mathrm{e}^{\sqrt{\mathrm{L}_F}t_k}\,\Big(\frac{1}{2}t_{k+1}t_{k+2}\varrho_A + \frac{1}{6}t_k t_{k+1}t_{k+2}\varrho_B + \frac{7}{6}t_k t_{k+1}t_{k+2}\varrho_A' + \frac{7}{24}t_k t_{k+1}t_{k+2}t_{k+3}\varrho_B' \\ &\qquad\qquad + \big(\frac{1}{6}(C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_k t_{k+1}t_{k+2} + \frac{1}{24}C_B^{\mathrm{LF}}t_{k-1}t_k t_{k+1}t_{k+2}\big)\tau^2\Big) \\ &= \Big(\varrho_A + t_{k+1}\varrho_B + 7t_k\varrho_A' + \frac{7}{2}t_{k+1}t_{k+2}\varrho_B' \\ &\quad + \mathrm{L}_F\,\mathrm{e}^{\sqrt{\mathrm{L}_F}t_k}\,\big(\frac{1}{2}t_{k+1}t_{k+2}\varrho_A + \frac{1}{6}t_k t_{k+1}t_{k+2}\varrho_B + \frac{7}{6}t_k t_{k+1}t_{k+2}\varrho_A' + \frac{7}{24}t_k t_{k+1}t_{k+2}t_{k+3}\varrho_B'\big)\Big) \\ &\quad + \tau^2\Big((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_{k+1} + \frac{1}{2}t_k t_{k+1}C_B^{\mathrm{LF}} \\ &\qquad\qquad + \mathrm{L}_F\,\mathrm{e}^{\sqrt{\mathrm{L}_F}t_k}\,\big(\frac{1}{6}(C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_k t_{k+1}t_{k+2} + \frac{1}{24}C_B^{\mathrm{LF}}t_{k-1}t_k t_{k+1}t_{k+2}\big)\Big).\end{aligned}$$

For $0 \le t_{k+3} \le T$ hence there are constants $C_A(T)$, $\widetilde{C}_A(T)$ depending on $\mathrm{L}_F, \varrho_A, \varrho_B, \varrho_A', \varrho_B', C_A^{\mathrm{LF}}$, and C_B^{LF} such that

$$\widetilde{E}_A^{k+1} \le C_A(T) + \tau^2\widetilde{C}_A(T).$$

By possibly increasing γ_A, we now assume that Assumption 4.5 holds for $\gamma_A \ge C_A(T) + \tau^2\widetilde{C}_A(T)$. Then we have $\widehat{E}_A^{k+1} \le 7\tau\varrho_A'$ by (4.31). Finally, we conclude from (4.36) and (4.19)

$$\begin{aligned}E_A^{k+1} &\le \varrho_A + t_{k+1}\varrho_B + \tau\mathrm{L}_F\sum_{j=0}^{k} t_{k+1-j}E_A^j + 7t_{k+1}\varrho_A' + \frac{7}{2}t_{k+1}t_{k+2}\varrho_B' \\ &\qquad\qquad + \left((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})t_{k+1} + \frac{1}{2}t_k t_{k+1}C_B^{\mathrm{LF}}\right)\tau^2.\end{aligned}$$

This completes the proof. □

With Lemma 4.10 we are now able to prove a global error bound.

Theorem 4.11. *If the assumptions of Lemma 4.10 are satisfied, then, as long as $t_{k+3} \leq T$, the global error E_A^{k+1} is bounded by*

$$E_A^{k+1} \leq \big(\varrho_A + T\varrho_B + 7T\varrho_A' + \frac{7}{2}T^2\varrho_B' + ((C_A^{\mathrm{LF}} + C_B^{\mathrm{LF}})T + \frac{1}{2}C_B^{\mathrm{LF}}T^2)\tau^2\big)\,\mathrm{e}^{\sqrt{\mathrm{L}_F}T}\,.$$

Proof. The bound for E_A^{k+1} is a direct consequence of (4.34) with $j = k+1$. □

The error of the St-LO scheme in Algorithm 7 is hence a combination of two error contributions: an error caused by the low-rank approximations on the one hand, and a time discretization error stemming from the leapfrog scheme.

Remark 4.12. If $r_{\mathbf{A}} = r_{\mathbf{B}} = \min\{m, n\}$, all constants $\varrho_A, \varrho_B, \varrho_A'$, and ϱ_B' vanish. We are then left with the time-discretization error, which coincides with the global error of the leapfrog scheme when it is applied to (4.1). This is no surprise, since by Theorem 4.3 we already know that in this situation the St-LO and leapfrog schemes are equivalent. ◇

In the proof of Theorem 4.11 we make use of the following Gronwall-type lemma.

Lemma 4.13. *Let $\tau, \mathrm{L} \geq 0$ and $\{M_k\}_{k\geq 0}$ a nonnegative, monotonically increasing sequence. If the nonnegative sequence $\{E_k\}_{k\geq 0}$ satisfies*

$$E_k \leq M_k + \tau^2\mathrm{L}\sum_{j=0}^{k-1}(k-j)E_j,$$

then

$$E_k \leq M_k\,\mathrm{e}^{\sqrt{\mathrm{L}}t_k}\,.$$

Proof. Define $\varepsilon_k := E_k/M_k$ for all $k \geq 0$. The sequence $\{\varepsilon_k\}_{k\geq 0}$ is nonnegative and satisfies

$$\varepsilon_k \leq 1 + \tau^2\mathrm{L}\sum_{j=0}^{k-1}(k-j)\frac{E_j}{M_j}\frac{M_j}{M_k} \leq 1 + \tau^2\mathrm{L}\sum_{j=0}^{k-1}(k-j)\varepsilon_j \tag{4.37}$$

due to the monotonicity of $\{M_k\}_{k\geq 0}$. Application of [Carle et al., 2020, Lemma 3.8] to (4.37) yields the bound

$$\varepsilon_k \leq \mathrm{e}^{\sqrt{\mathrm{L}}t_k},$$

and multiplication with M_k on both sides of the inequality hence the desired result. □

Similarly as for E_A^{k+1} we can derive a global error bound for $E_B^{k+\frac{1}{2}}$ based on the coupled recursions in Corollary 4.9. However, we will interchange the roles of the error contributions, i.e., we will now give representations of $E_B^{k+\frac{1}{2}}$ and E_A^k which only depend on the previous errors $E_B^{j+\frac{1}{2}}$, $j = 0, \ldots, k-1$, but not on E_A^j.

Lemma 4.14. *Let the assumptions of Lemma 4.10 be satisfied. Then for $k \geq 1$ such that $t_{k+3} \leq T$, the errors $E_B^{k+\frac12}$ and E_A^k satisfy*

$$E_A^k \leq \varrho_A + 7t_k\varrho_A' + \tau\sum_{j=0}^{k-1} E_B^{j+\frac12} + C_A^{\mathrm{LF}} t_k\tau^2, \tag{4.38a}$$

$$E_B^{k+\frac12} \leq E_B^{\frac12} + t_k\mathrm{L}_F\varrho_A + \frac72 t_kt_{k+1}\mathrm{L}_F\varrho_A' + 7t_k\varrho_B' + \tau^2\mathrm{L}_F\sum_{j=0}^{k-1}(k-j)E_B^{j+\frac12} + \big(\frac12 C_A^{\mathrm{LF}}\mathrm{L}_F t_kt_{k+1} + C_B^{\mathrm{LF}}t_k\big)\tau^2. \tag{4.38b}$$

Proof. We perform the proof again by induction. Furthermore, with the same argument as in the proof of Lemma 4.10, we will immediately bound the errors $\widehat{E}_A^k$ and $\widehat{E}_B^{k+\frac12}$ according to (4.31), since we will be again able to choose appropriate values for γ_A and γ_B.

For $k=0$, (4.20c) reads

$$\begin{aligned} E_A^1 &\leq E_A^0 + \widehat{E}_A^1 + \tau E_B^{\frac12} + C_A^{\mathrm{LF}}\tau^3 \\ &\leq \varrho_A + 7\tau\varrho_A' + \tau E_B^{\frac12} + C_A^{\mathrm{LF}}\tau^3, \end{aligned}$$

which is (4.38a) for $k=1$. Likewise, (4.25c) yields

$$\begin{aligned} E_B^{\frac32} &\leq (1+\tau^2\mathrm{L}_F)E_B^{\frac12} + \tau\mathrm{L}_F E_A^0 + \tau\mathrm{L}_F\widehat{E}_A^1 + \widehat{E}_B^{\frac32} + C_A^{\mathrm{LF}}\mathrm{L}_F\tau^4 + C_B^{\mathrm{LF}}\tau^3 \\ &\leq (1+\tau^2\mathrm{L}_F)E_B^{\frac12} + \tau\mathrm{L}_F\varrho_A + 7\tau^2\mathrm{L}_F\varrho_A' + 7\tau\varrho_B' + C_A^{\mathrm{LF}}\mathrm{L}_F\tau^4 + C_B^{\mathrm{LF}}\tau^3, \end{aligned}$$

and thus (4.38b) for $k=1$.

Assume now, that (4.38) holds for some $k-1 \in \mathbb{N}$. (4.20c) and the induction hypothesis yield

$$\begin{aligned} E_A^k &\leq E_A^{k-1} + \tau E_B^{k-\frac12} + \widehat{E}_A^k + C_A^{\mathrm{LF}}\tau^3 \\ &\leq \big(\varrho_A + 7t_{k-1}\varrho_A' + \tau\sum_{j=0}^{k-2} E_B^{j+\frac12} + C_A^{\mathrm{LF}}t_{k-1}\tau^2\big) + \tau E_B^{k-\frac12} + 7\tau\varrho_A' + C_A^{\mathrm{LF}}\tau^3 \\ &\leq \varrho_A + 7t_k\varrho_A' + \tau\sum_{j=0}^{k-1} E_B^{j+\frac12} + C_A^{\mathrm{LF}}t_k\tau^2. \end{aligned}$$

(4.25c) yields together with the induction hypothesis and (4.38a)

$$\begin{aligned} E_B^{k+\frac12} &\leq (1+\tau^2\mathrm{L}_F)E_B^{k-\frac12} + \tau\mathrm{L}_F E_A^{k-1} + \tau\mathrm{L}_F\widehat{E}_A^k + \widehat{E}_B^{k+\frac12} + C_A^{\mathrm{LF}}\mathrm{L}_F\tau^4 + C_B^{\mathrm{LF}}\tau^3 \\ &\leq \Big(E_B^{\frac12} + t_{k-1}\mathrm{L}_F\varrho_A + \frac72 t_{k-1}t_k\mathrm{L}_F\varrho_A' + 7t_{k-1}\varrho_B' + \tau^2\mathrm{L}_F\sum_{j=0}^{k-2}(k-1-j)E_B^{j+\frac12} \\ &\quad + \big(\frac12 C_A^{\mathrm{LF}}\mathrm{L}_F t_{k-1}t_k + C_B^{\mathrm{LF}}t_{k-1}\big)\tau^2\Big) \\ &\quad + \tau^2\mathrm{L}_F E_B^{k-\frac12} + \tau\mathrm{L}_F\Big(\varrho_A + 7t_{k-1}\varrho_A' + \tau\sum_{j=0}^{k-2} E_B^{j+\frac12} + C_A^{\mathrm{LF}}t_{k-1}\tau^2\Big) \\ &\quad + 7\tau^2\mathrm{L}_F\varrho_A' + 7\tau\varrho_B' + C_A^{\mathrm{LF}}\mathrm{L}_F\tau^4 + C_B^{\mathrm{LF}}\tau^3 \\ &\leq E_B^{\frac12} + t_k\mathrm{L}_F\varrho_A + \frac72 t_kt_{k+1}\mathrm{L}_F\varrho_A' + 7t_k\varrho_B' + \tau^2\mathrm{L}_F\sum_{j=0}^{k-1}(k-j)E_B^{j+\frac12} + \big(\frac12 C_A^{\mathrm{LF}}\mathrm{L}_F t_kt_{k+1} + C_B^{\mathrm{LF}}t_k\big)\tau^2. \end{aligned}$$

This completes the proof. □

Theorem 4.15. *If the assumptions of* Lemma 4.8 *are satisfied, then the global error* $E_B^{k+\frac{1}{2}}$ *is bounded by*

$$E_B^{k+\frac{1}{2}} \leq \mathrm{e}^{\sqrt{\mathrm{L}_F} t_k} M_B^{k+\frac{1}{2}},$$

for

$$M_B^{k+\frac{1}{2}} = \varrho_B + t_{k+\frac{1}{2}} \mathrm{L}_F \varrho_A + 7 t_{k+\frac{1}{2}} \varrho_B' + \frac{7}{2} t_k t_{k+1} \mathrm{L}_F \varrho_A' + \big(\frac{1}{2} C_A^{\mathrm{LF}} \mathrm{L}_F t_k t_{k+1} + (1+t_k) C_B^{\mathrm{LF}}\big)\tau^2.$$

Proof. We start by bounding the first $E_B^{\frac{1}{2}}$ in (4.38b) from above by (4.20a). This gives

$$\begin{aligned}
E_B^{k+\frac{1}{2}} &\leq \big(E_B^0 + \frac{\tau}{2}\mathrm{L}_F E_A^0 + \widehat{E}_B^{\frac{1}{2}} + C_B^{\mathrm{LF}}\tau^2\big) + t_k \mathrm{L}_F \varrho_A + \frac{7}{2} t_k t_{k+1} \mathrm{L}_F \varrho_A' + 7 t_k \varrho_B' + \tau^2 \mathrm{L}_F \sum_{j=0}^{k-1} (k-j) E_B^{j+\frac{1}{2}} \\
&\quad + \big(\frac{1}{2} C_A^{\mathrm{LF}} \mathrm{L}_F t_k t_{k+1} + C_B^{\mathrm{LF}} t_k\big)\tau^2 \\
&\leq \varrho_B + t_{k+\frac{1}{2}} \mathrm{L}_F \varrho_A + 7 t_{k+\frac{1}{2}} \varrho_B' + \frac{7}{2} t_k t_{k+1} \mathrm{L}_F \varrho_A' + \big(\frac{1}{2} C_A^{\mathrm{LF}} \mathrm{L}_F t_k t_{k+1} + (1+t_k) C_B^{\mathrm{LF}}\big)\tau^2 \\
&\quad + \tau^2 \mathrm{L}_F \sum_{j=0}^{k-1} (k-j) E_B^{j+\frac{1}{2}} \\
&= M_B^{k+\frac{1}{2}} + \tau^2 \mathrm{L}_F \sum_{j=0}^{k-1} (k-j) E_B^{j+\frac{1}{2}}.
\end{aligned}$$

Application of Lemma 4.13 then yields the stated result. □

So far, we can only provide the analysis of the St-LO scheme for the nonstiff case. In the following, we provide an outlook on how the stiff case might be tackled.

4.2.2 Outlook: Error analysis of the St-LO scheme for semilinear stiff problems

We now turn to the semilinear problem

$$A''(t) = -\Omega_1^2 A(t) - A(t)\Omega_2^2 + f\big(A(t)\big) \in \mathbb{C}^{m\times n}, \quad t \in [0,T], \qquad A(0) = A_0, \quad A'(0) = B_0, \tag{4.39}$$

where $\Omega_1^2 \in \mathbb{R}^{m\times m}$ and $\Omega_2^2 \in \mathbb{R}^{n\times n}$ are symmetric positive definite matrices with possibly large norm, and $f : \mathbb{C} \to \mathbb{C}$ is a globally Lipschitz continuous function with constant $\mathrm{L}_f > 0$, i.e.,

$$|f(a) - f(b)| \leq \mathrm{L}_f |a-b| \qquad \text{for all } a, b \in \mathbb{C}.$$

The evaluation of f at A is performed entrywise, i.e.,

$$f(A)_{ij} = f(A_{ij}), \qquad i = 1,\dots,m, \quad j = 1,\dots,n.$$

Since f is Lipschitz continuous, we have for all $A, B \in \mathbb{C}^{m\times n}$

$$\|f(A) - f(B)\|^2 = \sum_{i=1}^{m}\sum_{j=1}^{n} |f(A_{ij}) - f(B_{ij})|^2 \leq \sum_{i=1}^{m}\sum_{j=1}^{n} \mathrm{L}_f^2 |A_{ij} - B_{ij}|^2 = \mathrm{L}_f^2 \|A - B\|^2. \tag{4.40}$$

In [Carle et al., 2020] and recently in [Carle, 2021], a general class of two-step schemes for the numerical solution of the second-order semilinear differential equation

$$a''(t) = -\widetilde{L} a(t) + f\big(a(t)\big) \in \mathbb{C}^m, \quad t \in [0,T], \qquad a(0) = a_0, \quad a'(0) = b_0, \tag{4.41}$$

was studied. Here, $\widetilde{L} \in \mathbb{R}^{m\times m}$ is a symmetric positive definite matrix, and $f : \mathbb{C}^m \to \mathbb{C}^m$ is a globally Lipschitz continuous nonlinearity. The leapfrog scheme (4.5) is a special case of these general methods. In [Carle et al., 2020], the following result for the leapfrog scheme was derived:

Theorem 4.16 ([Carle et al., 2020, Theorem 4.3])**.** *Let $\vartheta \in (0, 1]$ and define the step size $\tau_{\mathrm{SSR}}^{(\vartheta)}$ via*

$$\tau_{\mathrm{SSR}}^{(\vartheta)} = \sqrt{\frac{4\vartheta^2}{\|\widetilde{L}\|_2}}. \tag{4.42}$$

Let $f : \mathbb{C}^m \to \mathbb{C}^m$ be globally Lipschitz continuous with constant L_f, i.e.,

$$\|f(a) - f(b)\|_2 \le \mathrm{L}_f \|a - b\|_2 \qquad \textit{for all } a, b \in \mathbb{C}^m.$$

In addition, assume that the exact solution a of (4.41) satisfies $a \in \mathcal{C}^4([0, T])$. Then, for $\tau \le \tau_{\mathrm{SSR}}^{(1)}$ and $t_k \le T$ we have for the approximations a_k of the scheme (4.5)

$$\|a(t_k) - a_k\|_2 \le (\frac{1}{2} C_1 T^2 + C_2 T)\, \mathrm{e}^{\sqrt{\mathrm{L}_f} T}\, \tau^2,$$

where the constants C_1, C_2 are independent of $\|\widetilde{L}\|_2$, k, and τ.

Observe that the norm of $\widetilde{L}$ causes a restriction on the step size, but the global error bound is independent of $\widetilde{L}$.

By application of the vectorization operator vec, which maps a matrix $A \in \mathbb{C}^{m\times n}$ to a vector $a \in \mathbb{C}^{mn}$ by stacking its columns upon each other, the matrix differential equation (4.39) is a special case of (4.41) with $\widetilde{L} = \Omega_1^2 \oplus \Omega_2^2 \in \mathbb{R}^{mn\times mn}$, cf. Definition A.15. By Definition A.10, Theorem A.11, and the symmetry of Ω_1^2 and Ω_2^2 it holds

$$\widetilde{L}^T = (\Omega_1^2 \oplus \Omega_2^2)^T = (I_n \otimes \Omega_1^2 + \Omega_2^2 \otimes I_m)^T = I_n \otimes (\Omega_1^2)^T + (\Omega_2^2)^T \otimes I_m = \widetilde{L},$$

so that $\widetilde{L}$ is symmetric. Since both Ω_1^2 and Ω_2^2 are positive definite matrices, we deduce from Theorem A.16 that the same is valid for $\widetilde{L}$. Moreover,

$$\|f(a) - f(b)\|_2 = \|f(A) - f(B)\| \le \mathrm{L}_f \|A - B\| = \mathrm{L}_f \|a - b\|_2, \qquad \text{for all } a, b \in \mathbb{C}^{mn}.$$

Using Theorem A.13, one can show that the leapfrog scheme applied to (4.41) yields the same approximations up to vectorization as if it is applied to (4.39). We therefore obtain the following result as an immediate consequence of Theorem 4.16:

Corollary 4.17. *Let $f : \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ satisfy (4.40) and assume that the exact solution A of (4.39) satisfies $A \in \mathcal{C}^4([0, T])$. Then, for $\tau \le \tau_{\mathrm{SSR}}^{(1)}$, where τ_{SSR} is given in (4.42) with $\widetilde{L} = \Omega_1^2 \oplus \Omega_2^2$, and $t_k \le T$ we have for the approximations A_k of the scheme (4.5)*

$$\|A(t_k) - A_k\| \le (\frac{1}{2} C_1 T^2 + C_2 T)\, \mathrm{e}^{\sqrt{\mathrm{L}_f} T}\, \tau^2,$$

where the constants C_1, C_2 are independent of $\|\widetilde{L}\|_2$, k and τ.

Since it is possible to derive a bound on the global error of the leapfrog scheme which does not involve the norms of the matrices Ω_1^2 and Ω_2^2 we are optimistic that the same is possible for the St-LO scheme. In the spirit of Section 4.2.1 we expect that the global error of the St-LO scheme admits a bound which is of the same form as in Corollary 4.17, with additional low-rank error contributions.

In the following, we adapt the technique which was used in [Carle et al., 2020] and [Carle, 2021] to prove Theorem 4.16 and thus work with the vectorized formulation of (4.39) throughout. We abbreviate

$$\mathbf{a}_k = \operatorname{vec} \mathbf{A}_k, \qquad \mathbf{b}_{k-\frac{1}{2}} = \operatorname{vec} \mathbf{B}_{k-\frac{1}{2}},$$

for the low-rank approximations $\mathbf{A}_k \approx A(t_k)$ and $\mathbf{B}_{k-\frac{1}{2}} \approx A'(t_{k-\frac{1}{2}})$ to the exact solution of (4.39) and its derivative, respectively. Furthermore, we will make multiple use of the identity

$$\|A\| = \|\operatorname{vec} A\|_2 \qquad \text{for all } A \in \mathbb{C}^{m\times n},$$

without always indicating this explicitly. For $A \in \mathcal{C}^\ell([0,T])$, $\ell \in \mathbb{N}$, we abbreviate bounds on the ℓth derivative of A by

$$N_t^{(\ell)} = \max_{0\le\xi\le t} \|A^{(\ell)}(\xi)\|, \qquad \ell = 1, 2, \dots.$$

Besides, we denote by

$$\mathbf{d}_k^{(\ell),\pm} = \int_{t_k}^{t_{k\pm1}} \frac{(t_{k\pm1}-\xi)^\ell}{\ell!} A^{(\ell+1)}(\xi)\,\mathrm{d}\xi$$

the remainder terms of the ℓth order Taylor expansion of $A(t_{k\pm1})$ at t_k. By Theorem 4.6, they are bounded by

$$\|\mathbf{d}_k^{(\ell),+}\| \le \frac{\tau^{\ell+1}}{(\ell+1)!} \max_{t_k\le t\le t_{k+1}} \|A^{(\ell+1)}(t)\|, \qquad \|\mathbf{d}_k^{(\ell),-}\| \le \frac{\tau^{\ell+1}}{(\ell+1)!} \max_{t_{k-1}\le t\le t_k} \|A^{(\ell+1)}(t)\|.$$

We continue by stating two results which are required for the analysis. The first one allows us to write the kth approximation of the leapfrog scheme explicitly in terms of the starting values a_0 and a_1, and the nonlinearity f evaluated at previous approximations:

Theorem 4.18 ([Carle, 2021, Theorem 3.18])**.** *Let $\tau \le \tau_{\mathrm{SSR}}^{(1)}$. For the approximations of the two-step scheme (4.5) with starting value $a_1 = a_0 + \tau b_0 + \frac{\tau^2}{2} f(a_0)$ applied to (4.41) we have*

$$a_k = \cos(k\mathbf{\Phi})a_0 + \mathcal{S}_k(a_1 - \cos\mathbf{\Phi}a_0) + \tau^2 \sum_{j=1}^{k-1} \mathcal{S}_{k-j} f(a_j), \qquad \mathcal{S}_\ell = \frac{\sin(\ell\mathbf{\Phi})}{\sin\mathbf{\Phi}} \ (\ell \in \mathbb{N}_0),$$

where $\mathbf{\Phi} \in \mathbb{R}^{m\times m}$ is a symmetric matrix with spectrum in $[0,\pi]$ which is uniquely defined by

$$\cos\mathbf{\Phi} = I - \frac{1}{2}\tau^2\widetilde{L} \qquad \textit{and satisfies} \qquad \sin\mathbf{\Phi} = \left(\tau^2\widetilde{L}(I - \frac{1}{4}\tau^2\widetilde{L})\right)^{\frac{1}{2}}.$$

For the following, we need a bound of the difference $\mathcal{S}_{\ell+1} - \mathcal{S}_\ell$.

Lemma 4.19. *Let $\mathcal{S}_\ell$ be defined as in Theorem 4.18 and $\vartheta \in (0,1)$. Then it holds for every $\ell \in \mathbb{N}_0$*

$$\|\mathcal{S}_{\ell+1} - \mathcal{S}_\ell\|_2 \le \begin{cases} 2\ell+1, & \tau \le \tau_{\mathrm{SSR}}^{(1)}, \\ \dfrac{1}{\sqrt{1-\vartheta^2}}, & \tau \le \tau_{\mathrm{SSR}}^{(\vartheta)}. \end{cases} \tag{4.43}$$

Proof. In [Carle, 2021, Lemma 3.23 and Example 3.26] it was shown that the norm of the matrices $\mathcal{S}_\ell$ is bounded by

$$\tau\|\mathcal{S}_\ell\|_2 \le \begin{cases} \tau\ell, & \tau \le \tau_{\mathrm{SSR}}^{(1)}, \\ \dfrac{\|\widetilde{L}^{-\frac{1}{2}}\|_2}{\sqrt{1-\vartheta^2}}, & \tau \le \tau_{\mathrm{SSR}}^{(\vartheta)}, \quad \vartheta \in (0,1). \end{cases} \tag{4.44}$$

The first bound in (4.43) is a direct consequence of (4.44). For the second, recall that $\widetilde{L}$ is real and symmetric, and therefore orthogonally diagonalizable. It hence suffices to study the eigenvalues ϕ of $\mathbf{\Phi}$. Using the trigonometric identities

$$\sin\theta - \sin\varphi = 2\sin\left(\frac{\theta-\varphi}{2}\right)\cos\left(\frac{\theta+\varphi}{2}\right) \qquad \text{and} \qquad \sin(2\theta) = 2\sin(\theta)\cos(\theta), \qquad \theta,\varphi\in\mathbb{R},$$

we obtain

$$\left|\frac{\sin\big((\ell+1)\phi\big)-\sin(\ell\phi)}{\sin\phi}\right| = \left|\frac{2\sin\left(\frac{\phi}{2}\right)\cos\left(\frac{2\ell+1}{2}\phi\right)}{\sin\phi}\right| = \left|\frac{\cos\left(\frac{2\ell+1}{2}\phi\right)}{\cos\frac{\phi}{2}}\right| \le \left|\frac{1}{\cos\frac{\phi}{2}}\right| \le \frac{1}{\sqrt{1-\vartheta^2}}.$$

The last inequality follows from the proof of [Carle, 2021, Lemma 3.23(b)]. □

In the next lemma, we derive an explicit representation of the global error between the exact solution of (4.41) and its low-rank approximation computed with the St-LO scheme. Inserting the low-rank approximations into the leapfrog update steps yields the defects δ_A^{k+1} and $\delta_B^{k+\frac12}$ defined via

$$\mathbf{B}_{k+\frac12} = \mathbf{B}_{k-\frac12} + \tau(-\Omega_1^2\mathbf{A}_k - \mathbf{A}_k\Omega_2^2 + f(\mathbf{A}_k)) + \delta_B^{k+\frac12}, \tag{4.45a}$$

$$\mathbf{A}_{k+1} = \mathbf{A}_k + \tau\mathbf{B}_{k+\frac12} + \delta_A^{k+1}. \tag{4.45b}$$

By Section 4.2.1 they can also be written as

$$\delta_A^{k+1} = \widetilde{A}_k(\tau) - \mathbf{A}_{k+1} \qquad \text{and} \qquad \delta_B^{k+\frac12} = \widetilde{B}_{k-\frac12}(\tau) - \mathbf{B}_{k+\frac12},$$

where $\widetilde{A}_k$ and $\widetilde{B}_{k-\frac12}$ are given in (4.7).

Lemma 4.20. *The global error $\mathbf{e}_k = a(t_k) - \mathbf{a}_k$ between the exact solution of (4.41) and its low-rank approximation computed with the St-LO scheme satisfies*

$$\mathbf{e}_1 = (I - \frac{\tau^2}{2}\widetilde{L})\mathbf{e}_0 + \tau(b_0 - \mathbf{b}_0) + \frac{\tau^2}{2}\big(f(a_0) - f(\mathbf{a}_0)\big) + \operatorname{vec}\big(\boldsymbol{\delta} - \tau\delta_B^{\frac12} - (\delta_A^1 - \delta_A^0)\big), \tag{4.46}$$

and

$$\begin{aligned}\mathbf{e}_k = {} & \cos(k\mathbf{\Phi})\mathbf{e}_0 + \tau\mathcal{S}_k(b_0 - \mathbf{b}_0) + \tau^2\sum_{j=0}^{k-1}\zeta_j\mathcal{S}_{k-j}\big(f(a(t_j)) - f(\mathbf{a}_j)\big) + \mathcal{S}_k\operatorname{vec}\boldsymbol{\delta} + \sum_{j=1}^{k-1}\mathcal{S}_{k-j}\operatorname{vec}\mathbf{d}_j \\ & - \tau\sum_{j=0}^{k-1}\mathcal{S}_{k-j}\operatorname{vec}\delta_B^{j+\frac12} - \sum_{j=0}^{k-1}\mathcal{S}_{k-j}\operatorname{vec}(\delta_A^{j+1} - \delta_A^j), \qquad k>1.\end{aligned} \tag{4.47}$$

Proof. We show that the global error satisfies the leapfrog recurrence (4.5) up to a defect. By inserting the exact solution of (4.41) into (4.5), one gets as in [Carle, 2021, Section 3.4]

$$a(t_{k+1}) - 2a(t_k) + a(t_{k-1}) = -\tau^2\widetilde{L}a(t_k) + \tau^2 f\big(a(t_k)\big) + \operatorname{vec}\mathbf{d}_k,$$

and

$$a(\tau) = (I - \frac{\tau^2}{2}\widetilde{L})a_0 + \tau b_0 + \frac{\tau^2}{2}f(a_0) + \operatorname{vec}\boldsymbol{\delta}.$$

The defects $\mathbf{d}_k$ and $\boldsymbol{\delta}$ are bounded by

$$\|\mathbf{d}_k\| \le \frac{\tau^4}{24}\max_{t\in[t_k,t_{k+1}]}\|A^{(4)}(t)\| + \frac{\tau^4}{24}\max_{t\in[t_{k-1},t_k]}\|A^{(4)}(t)\| \le C_1\tau^4,$$

and

$$\|\boldsymbol{\delta}\| \leq \frac{\tau^3}{8} N^{(3)}_{\frac{\tau}{2}} + \frac{\tau^3}{24} N^{(3)}_{\tau} \leq C_2 \tau^3,$$

respectively. From (4.45) we have for $k \geq 1$

$$\begin{aligned}
\mathbf{a}_{k+1} - 2\mathbf{a}_k + \mathbf{a}_{k-1} &= (\mathbf{a}_{k+1} - \mathbf{a}_k) - (\mathbf{a}_k - \mathbf{a}_{k-1}) \\
&= \tau \mathbf{b}_{k+\frac{1}{2}} + \operatorname{vec} \delta_A^{k+1} - \tau \mathbf{b}_{k-\frac{1}{2}} - \operatorname{vec} \delta_A^k \\
&= \tau\big(\mathbf{b}_{k-\frac{1}{2}} + \tau(-\widetilde{L}\mathbf{a}_k + f(\mathbf{a}_k)) + \operatorname{vec} \delta_B^{k+\frac{1}{2}}\big) + \operatorname{vec} \delta_A^{k+1} - \tau \mathbf{b}_{k-\frac{1}{2}} - \operatorname{vec} \delta_A^k \\
&= -\tau^2 \widetilde{L}\mathbf{a}_k + \tau^2 f(\mathbf{a}_k) + \operatorname{vec}\big(\tau \delta_B^{k+\frac{1}{2}} + (\delta_A^{k+1} - \delta_A^k)\big),
\end{aligned}$$

and the approximations $\mathbf{a}_k$ satisfy the two-step leapfrog recurrence up to a defect given in terms of the defects δ_A^{k+1}, δ_A^k, and $\delta_B^{k+\frac{1}{2}}$, respectively. For $k = 0$ it is

$$\begin{aligned}
\mathbf{a}_1 &= \mathbf{a}_0 + \tau \mathbf{b}_{\frac{1}{2}} + \operatorname{vec} \delta_A^1 \\
&= \mathbf{a}_0 + \tau\big(\mathbf{b}_0 + \frac{\tau}{2}(-\widetilde{L}\mathbf{a}_0 + f(\mathbf{a}_0)) + \operatorname{vec} \delta_B^{\frac{1}{2}}\big) + \operatorname{vec} \delta_A^1 \\
&= (I - \frac{\tau^2}{2}\widetilde{L})\mathbf{a}_0 + \tau \mathbf{b}_0 + \frac{\tau^2}{2} f(\mathbf{a}_0) + \operatorname{vec}\big(\tau \delta_B^{\frac{1}{2}} + \delta_A^1 - \delta_A^0\big),
\end{aligned}$$

where $\delta_A^0 = 0$. Overall, the error $\mathbf{e}_k = a(t_k) - \mathbf{a}_k$ satisfies the two-step recursion

$$\mathbf{e}_{k+1} - 2\mathbf{e}_k + \mathbf{e}_{k-1} = -\tau^2 \widetilde{L}\mathbf{e}_k + \tau^2\big(f(a(t_k) - f(\mathbf{a}_k)\big) + \operatorname{vec}\big(\mathbf{d}_k - \tau\delta_B^{k+\frac{1}{2}} - (\delta_A^{k+1} - \delta_A^k)\big), \qquad k > 1.$$

For $k = 1$ it is

$$\mathbf{e}_1 = (I - \frac{\tau^2}{2}\widetilde{L})\mathbf{e}_0 + \tau(b_0 - \mathbf{b}_0) + \frac{\tau^2}{2}\big(f(a_0) - f(\mathbf{a}_0)\big) + \operatorname{vec}\big(\boldsymbol{\delta} - \tau\delta_B^{\frac{1}{2}} - (\delta_A^1 - \delta_A^0)\big),$$

hence (4.46). Overall, $\mathbf{e}_k$ satisfies (4.5) up to a defect. Application of Theorem 4.18 then yields the representation (4.47) of $\mathbf{e}_k$. □

In [Carle, 2021], a bound on the error $\mathbf{e}_k$ was derived by bounding all contributions on the right-hand side of (4.47) individually, and making use of Lemma 4.13 afterwards. However, it was mentioned that for a uniform bound in k the defects need to be bounded like $C\tau^2$ for some $C > 0$. By the above, this is true for the defects $\boldsymbol{\delta}$ and $\mathbf{d}_j$. For the defects $\delta_B^{j+\frac{1}{2}}$ and δ_A^j we can relax this condition to bounds like

$$\|\delta_A^k\| \leq C\tau, \quad \|\delta_B^{k+\frac{1}{2}}\| \leq C\tau, \qquad C > 0, \tag{4.48}$$

and still obtain a uniform bound in k: Suppose (4.48) is satisfied for all k with $t_{k+1} \leq T$. By (4.44) it holds for $\tau \leq \tau_{\mathrm{SSR}}^{(1)}$

$$\|\tau \sum_{j=0}^{k-1} \mathcal{S}_{k-j} \operatorname{vec} \delta_B^{j+\frac{1}{2}}\|_2 \leq \tau \sum_{j=0}^{k-1} \|\mathcal{S}_{k-j}\|_2 \, \|\delta_B^{j+\frac{1}{2}}\| \leq C\tau^2 \sum_{j=0}^{k-1}(k-j) \leq \frac{1}{2} C t_k t_{k+1} \leq \frac{1}{2} C T^2.$$

Using $\delta_A^0 = 0$, the last sum in (4.47) can be rewritten into

$$\sum_{j=0}^{k-1} \mathcal{S}_{k-j} \operatorname{vec}(\delta_A^{j+1} - \delta_A^j) = \mathcal{S}_1 \operatorname{vec} \delta_A^k + \sum_{j=1}^{k-1}(\mathcal{S}_{j+1} - \mathcal{S}_j) \operatorname{vec} \delta_A^{k-j}.$$

Under the stronger step size restriction $\tau \leq \tau_{\mathrm{SSR}}^{(\vartheta)}$, $\vartheta \in (0,1)$, we can make use of the second bound in (4.43) to obtain

$$\begin{aligned}\Big\| \sum_{j=0}^{k-1} \mathcal{S}_{k-j} \operatorname{vec}(\delta_A^{j+1} - \delta_A^{j}) \Big\|_2 &\leq \|\mathcal{S}_1 \operatorname{vec} \delta_A^{k}\|_2 + \sum_{j=1}^{k-1} \|(\mathcal{S}_{j+1} - \mathcal{S}_j) \operatorname{vec} \delta_A^{k-j}\|_2 \\ &\leq C\tau + \sum_{j=1}^{k-1} \frac{1}{\sqrt{1-\vartheta^2}} C\tau \\ &\leq CT \frac{1}{\sqrt{1-\vartheta^2}}.\end{aligned}$$

Unfortunately, so far there are no analytical results yielding the bounds (4.48) independent of $\widetilde{L}$. Thus, it is not yet possible to derive a global error bound for the St-LO scheme which is of similar form as the bound in Corollary 4.17, and is fully independent of the norms of Ω_1^2 and Ω_2^2. This issue will be addressed in the future.

4.3 Stiff problems

We now develop a dynamical low-rank integrator for semilinear second-order matrix differential equations of the form

$$A''(t) = -\Omega_1^2 A(t) - A(t)\Omega_2^2 + f\big(A(t)\big), \quad t \in [0,T], \qquad A(0) = A_0, \quad A'(0) = B_0, \tag{4.49}$$

see also (4.39). In contrast to Section 4.2.2, here $\Omega_1 \in \mathbb{C}^{m\times m}$ and $\Omega_2 \in \mathbb{C}^{n\times n}$ are constant, positive semidefinite matrices. The nonlinearity f is Lipschitz continuous with moderate Lipschitz constant. It was shown, that though the global error of the leapfrog scheme (4.5) applied to (4.49) is independent of the matrices Ω_1 and Ω_2, they induce a restriction on the step size τ, cf. Corollary 4.17. Due to the strong connection between the leapfrog and the St-LO scheme, we suspect that the same holds for the latter. However, this is unsatisfactory in the situation, where (4.49) originates from a space-discretized wave-type equation. Then the linear part of the right-hand side corresponds to the discretization of the Laplacian. If the resolution in space is fine, the norms of Ω_1 and Ω_2 become large and cause a severe restriction on the step size. The objective of this section is to derive a variant of the St-LO scheme which exploits the structure of the right-hand side and allows for larger step sizes than the original ansatz. This is achieved by performing an additional splitting.

The construction of the dynamical low-rank integrator tailored to the problem (4.49) is based on the first-order formulation

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} B(t) \\ -\Omega_1^2 A(t) - A(t)\Omega_2^2 + f(A(t)) \end{bmatrix} = \begin{bmatrix} \omega_1^2 B(t) \\ -\Omega_1^2 A(t) \end{bmatrix} + \begin{bmatrix} \omega_2^2 B(t) \\ -A(t)\Omega_2^2 \end{bmatrix} + \begin{bmatrix} \omega_3^2 B(t) \\ f\big(A(t)\big) \end{bmatrix},$$

where we introduced weights $\omega_j \geq 0$, $j = 1,2,3$, satisfying

$$\omega_1^2 + \omega_2^2 + \omega_3^2 = 1,$$

see also Appendix B. The split equations then read

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} \omega_1^2 B(t) \\ -\Omega_1^2 A(t) \end{bmatrix} = \begin{bmatrix} 0 & \omega_1^2 I \\ -\Omega_1^2 & 0 \end{bmatrix} \begin{bmatrix} A(t) \\ B(t) \end{bmatrix}, \tag{4.50a}$$

$$\begin{bmatrix} A(t) & B(t) \end{bmatrix}' = \begin{bmatrix} \omega_2^2 B(t) & -A(t)\Omega_2^2 \end{bmatrix} = \begin{bmatrix} A(t) & B(t) \end{bmatrix} \begin{bmatrix} 0 & -\Omega_2^2 \\ \omega_2^2 I & 0 \end{bmatrix}, \tag{4.50b}$$

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} \omega_3^2 B(t) \\ f\big(A(t)\big) \end{bmatrix}. \tag{4.50c}$$

We denote the exact flows (cf. Definition B.1) of the three subproblems in (4.50) by $\varphi^{[\Omega_1]}$, $\varphi^{[\Omega_2]}$, and $\varphi^{[f]}$, respectively. A splitting method for solving (4.49) can be derived by applying theses flows in a symmetric way, which gives

$$\phi_\tau = \varphi_{\frac{\tau}{2}}^{[\Omega_1]} \circ \varphi_{\frac{\tau}{2}}^{[\Omega_2]} \circ \varphi_{\tau}^{[f]} \circ \varphi_{\frac{\tau}{2}}^{[\Omega_2]} \circ \varphi_{\frac{\tau}{2}}^{[\Omega_1]}. \tag{4.51}$$

In order to construct a low-rank scheme, we follow the idea in [Ostermann et al., 2019], see also Section 3.3, and replace all flows by low-rank flows in the splitting scheme (4.51). As the last subproblem (4.50c) is of the form (4.12), a low-rank approximation to the exact solution is readily obtained by the variant of the St-LO scheme presented in Algorithm 8. Hence, we replace the flow $\varphi^{[f]}$ by the numerical flow of Algorithm 8, which we denote by $\phi^{[\mathtt{stlo}]}$. It thus remains to replace the flows $\varphi^{[\Omega_1]}$ and $\varphi^{[\Omega_2]}$ appropriately by low-rank flows.

Consider the subproblem (4.50a) first, where the initial values to the problem are low-rank matrices $\mathbf{A}_0 \in \mathcal{M}_{r_\mathbf{A}}$ and $\mathbf{B}_0 \in \mathcal{M}_{r_\mathbf{B}}$. The exact solution to (4.50a) is

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix} = \exp\left(t \begin{bmatrix} 0 & \omega_1^2 I \\ -\Omega_1^2 & 0 \end{bmatrix} \right) \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 \end{bmatrix} = \begin{bmatrix} \cos(\omega_1 t \Omega_1) & \omega_1^2 t \operatorname{sinc}(\omega_1 t \Omega_1) \\ -t\Omega_1^2 \operatorname{sinc}(\omega_1 t \Omega_1) & \cos(\omega_1 t \Omega_1) \end{bmatrix} \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 \end{bmatrix}, \tag{4.52}$$

cf. Section A.5. By Theorem A.2, the ranks of A and A' satisfy

$$\operatorname{rank} A(t), \operatorname{rank} A'(t) \le \operatorname{rank} \mathbf{A}_0 + \operatorname{rank} \mathbf{B}_0 = r_\mathbf{A} + r_\mathbf{B} \qquad \text{for all } t \ge 0,$$

so that the exact flow $\varphi^{[\Omega_1]}$ of (4.50a) does not preserve the ranks of its input values in general. Yet, the exact solution to the problem is known in closed form, so we can compute low-rank approximations to A and B by the projector-splitting integrator with increments, cf. Algorithm 2. Note that the modified routine (4.14) can be employed to compute $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H \approx A(\tau)$ and $\mathbf{B}_1 = \mathbf{T}_1\mathbf{R}_1\mathbf{W}_1^H \approx A'(\tau)$, since in both cases the algorithm is started from a low-rank initial value.

For (4.50b), it is convenient to look at the conjugate transpose of the differential equation,

$$\begin{bmatrix} A^H(t) \\ B^H(t) \end{bmatrix}' = \begin{bmatrix} 0 & \omega_2^2 I \\ -(\Omega_2^2)^H & 0 \end{bmatrix} \begin{bmatrix} A^H(t) \\ B^H(t) \end{bmatrix}, \qquad \begin{bmatrix} A^H(0) \\ B^H(0) \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0^H \\ \mathbf{B}_0^H \end{bmatrix}.$$

As in (4.52),

$$\begin{bmatrix} A^H(t) \\ B^H(t) \end{bmatrix} = \begin{bmatrix} \cos(\omega_2 t \Omega_2^H) & \omega_2^2 t \operatorname{sinc}(\omega_2 t \Omega_2^H) \\ -t(\Omega_2^2)^H \operatorname{sinc}(\omega_2 t \Omega_2^H) & \cos(\omega_2 t \Omega_2^H) \end{bmatrix} \begin{bmatrix} \mathbf{A}_0^H \\ \mathbf{B}_0^H \end{bmatrix},$$

or equivalently

$$\begin{aligned}\begin{bmatrix} A(t) & B(t) \end{bmatrix} &= \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 \end{bmatrix} \begin{bmatrix} \cos(\omega_2 t\Omega_2^H)^H & -t\Omega_2^2 \operatorname{sinc}(\omega_2 t\Omega_2^H)^H \\ \omega_2^2 t \operatorname{sinc}(\omega_2 t\Omega_2^H)^H & \cos(\omega_2 t\Omega_2^H)^H \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 \end{bmatrix} \begin{bmatrix} \cos(\omega_2 t\Omega_2) & -t\Omega_2^2 \operatorname{sinc}(\omega_2 t\Omega_2) \\ \omega_2^2 t \operatorname{sinc}(\omega_2 t\Omega_2) & \cos(\omega_2 t\Omega_2) \end{bmatrix}\end{aligned}$$

by Theorem A.21. We can again compute a low-rank approximation to the exact solutions A and B by the recurrence displayed in (4.14).

With Remark 4.1, the numerical flows of computing low-rank approximations to the exact solutions of (4.50a) and (4.50b) by the modified routine (4.14) are given as

$$\phi_\tau^{[\Omega_1]} = \phi_\tau^{[\texttt{prsi}]} \circ \varphi_\tau^{[\Omega_1]}, \qquad \text{and} \qquad \phi_\tau^{[\Omega_2]} = \phi_\tau^{[\texttt{prsi}]} \circ \varphi_\tau^{[\Omega_2]},$$

respectively. Hence, we define the numerical flow of a dynamical low-rank integrator tailored to the stiff second-order problem (4.49) as

$$\phi_\tau^{[\texttt{stlostiff}]} = \phi_{\frac{\tau}{2}}^{[\Omega_1]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_2]} \circ \phi_\tau^{[\texttt{stlo}]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_2]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_1]}.$$

One single step of the method, which we call `stlostiff`, is then given as

$$\begin{bmatrix} \mathbf{A}_{k+1} \\ \mathbf{B}_{k+1} \end{bmatrix} = \phi_\tau^{[\texttt{stlostiff}]} \begin{bmatrix} \mathbf{A}_k \\ \mathbf{B}_k \end{bmatrix}, \qquad k = 0, 1, \dots,$$

and the overall scheme reads

$$\begin{bmatrix} \mathbf{A}_{k+1} \\ \mathbf{B}_{k+1} \end{bmatrix} = \left(\phi_\tau^{[\texttt{stlostiff}]}\right)^{k+1} \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 \end{bmatrix}. \tag{4.53}$$

A computationally cheaper variant is

$$\begin{bmatrix} \mathbf{A}_{k+1} \\ \mathbf{B}_{k+1} \end{bmatrix} = \phi_{\frac{\tau}{2}}^{[\Omega_1]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_2]} \circ \phi_\tau^{[\texttt{stlo}]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_2]} \circ \left(\phi_\tau^{[\Omega_1]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_2]} \circ \phi_\tau^{[\texttt{stlo}]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_2]}\right)^k \circ \phi_{\frac{\tau}{2}}^{[\Omega_1]} \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 \end{bmatrix}, \tag{4.54}$$

which is obtained by formally combining the last substep of the kth full step with the first substep of the subsequent full step. Note that the approximations $\mathbf{A}_{k+1}$ and $\mathbf{B}_{k+1}$ computed with the scheme (4.53) and its variant (4.54) will not coincide, since in general

$$\phi_{\frac{\tau}{2}}^{[\Omega_1]} \circ \phi_{\frac{\tau}{2}}^{[\Omega_1]} = \phi_{\frac{\tau}{2}}^{[\texttt{prsi}]} \circ \varphi_{\frac{\tau}{2}}^{[\Omega_1]} \circ \phi_{\frac{\tau}{2}}^{[\texttt{prsi}]} \circ \varphi_{\frac{\tau}{2}}^{[\Omega_1]} \neq \phi_\tau^{[\texttt{prsi}]} \circ \varphi_\tau^{[\Omega_1]} = \phi_\tau^{[\Omega_1]},$$

as the flows $\phi^{[\texttt{prsi}]}$ and $\varphi^{[\Omega_1]}$ do not commute.

A special situation occurs if the weight ω_i of the ith subproblem in (4.50) is zero. For $\omega_1 = 0$, we have

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix} = \begin{bmatrix} I_m & 0 \\ -t\Omega_1^2 & I_m \end{bmatrix} \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 - t\Omega_1^2 \mathbf{A}_0 \end{bmatrix}.$$

If $\omega_2 = 0$, the solution to the second subproblem (4.50b) reads

$$\begin{bmatrix} A(t) & B(t) \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 \end{bmatrix} \begin{bmatrix} I_n & -t\Omega_2^2 \\ 0 & I_n \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 - t\mathbf{A}_0\Omega_2^2 \end{bmatrix},$$

and for $\omega_3 = 0$ we have for the solution to the last subproblem (4.50c)

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 + tf(\mathbf{A}_0) \end{bmatrix}.$$

None of these solutions involve matrix functions of Ω_1 or Ω_2, and all problems can be solved exactly with only small computational effort. We therefore treat these cases separately in the implementation of the `stlostiff` scheme.

Remark 4.21. It may appear attractive to perform a splitting like

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} B(t) \\ -\Omega_1^2 A(t) - A(t)\Omega_2^2 + f(A(t)) \end{bmatrix} = \begin{bmatrix} \widetilde{\omega}_1^2 B(t) \\ -\Omega_1^2 A(t) - A(t)\Omega_2^2 \end{bmatrix} + \begin{bmatrix} \widetilde{\omega}_2^2 B(t) \\ f(A(t)) \end{bmatrix}, \qquad \widetilde{\omega}_1^2 + \widetilde{\omega}_2^2 = 1, \tag{4.55}$$

instead of (4.50). The nonlinear subproblem can be treated with the variant Algorithm 8 of the St-LO scheme. However, the linear subproblem cannot be implemented (storage) efficiently (see the discussion in Section 5.4), which is why we do not pursue this approach further. ◇

4.4 Highly oscillatory problems

In this section, we design a dynamical low-rank integrator for a special case of the second-order problem (4.1), namely

$$A''(t) = -\Omega^2 A(t) + f\big(A(t)\big) \in \mathbb{C}^{m\times n}, \qquad t \in [0,T], \qquad A(0) = A_0, \quad A'(0) = B_0, \tag{4.56}$$

with a symmetric, positive definite matrix Ω with arbitrary large norm $\omega = \|\Omega\|_2$. The nonlinear right-hand side f is assumed to be Lipschitz continuous. By the variation-of-constants formula we have

$$\begin{bmatrix} A(t+\tau) \\ A'(t+\tau) \end{bmatrix} = \begin{bmatrix} \cos(\tau\Omega) & \tau\,\mathrm{sinc}(\tau\Omega) \\ -\tau\Omega^2\,\mathrm{sinc}(\tau\Omega) & \cos(\tau\Omega) \end{bmatrix} \begin{bmatrix} A(t) \\ A'(t) \end{bmatrix} + \int_0^\tau \begin{bmatrix} \Omega^{-1}\sin\big((\tau-\sigma)\Omega\big) \\ \cos\big((\tau-\sigma)\Omega\big) \end{bmatrix} f\big(A(t+\sigma)\big)\,\mathrm{d}\sigma. \tag{4.57}$$

Such problems are often highly oscillatory: Due to the large eigenvalues of Ω the solution of (4.56) oscillates heavily. These oscillations are challenging in the construction of numerical integrators for (4.56).

In the following, we first revise trigonometric integrators for the numerical approximation to (4.57) before we construct a dynamical low-rank scheme.

4.4.1 Gautschi-type integrators

For second-order problems, the leapfrog scheme from Section 4.1 is considered the standard integration scheme. However, for problems of type (4.56), it suffers from a step size restriction due to stability issues, cf. Section 4.2.2. Even for the linear case $f \equiv 0$, the step sizes τ have to be chosen such that $\tau\omega < 2$ is satisfied, where $\omega = \|\Omega\|_2$ is again the largest frequency of the linear problem, cf. [Hairer et al., 2006, Section XIII.1].

In [Gautschi, 1961], a method for computing a numerical solution to (4.56) was proposed. It gives the exact solution for arbitrary Ω if $f \equiv \text{const}$. A first modification of this ansatz was introduced in [Deuflhard, 1979], and numerous people worked on improvements of the original method. All these methods share the following one-step formulation which is motivated by the representation (4.57) of the exact solution, cf. [Hairer et al., 2006, Section XIII.2.2]:

$$A_{k+1} = \cos(\tau\Omega)A_k + \tau \operatorname{sinc}(\tau\Omega)B_k + \frac{\tau^2}{2}\Psi f_k, \tag{4.58a}$$

$$B_{k+1} = -\tau\Omega^2 \operatorname{sinc}(\tau\Omega)A_k + \cos(\tau\Omega)B_k + \frac{\tau}{2}(\Psi_0 f_k + \Psi_1 f_{k+1}). \tag{4.58b}$$

Here, we used the abbreviations

$$f_k = f(\Phi A_k), \quad \Psi = \psi(\tau\Omega), \quad \Psi_0 = \psi_0(\tau\Omega), \quad \Psi_1 = \psi_1(\tau\Omega), \quad \Phi = \phi(\tau\Omega), \tag{4.59}$$

with even, real valued *filter functions* ψ, ψ_0, ψ_1, and ϕ satisfying $\psi(0) = \psi_0(0) = \psi_1(0) = \phi(0) = 1$. Hence, the methods involve (trigonometric) functions of the matrix Ω.

The purpose of the filter functions (4.59) is to give a substantial improvement over the original method introduced by Gautschi. They have to be chosen depending on the desired properties of the scheme. In the following, we choose the functions as in [Grimm and Hochbruck, 2006],

$$\psi(\xi) = \operatorname{sinc}^3(\xi), \qquad \psi_0(\xi) = \cos(\xi)\operatorname{sinc}^2(\xi), \qquad \psi_1(\xi) = \operatorname{sinc}^2(\xi), \qquad \phi(\xi) = \operatorname{sinc}(\xi), \tag{4.60}$$

which yields a symmetric scheme conserving the energy up to order τ. For simplicity, we will refer to the method (4.58) with the choice (4.60) of filter functions as *Gautschi method*, though it is in fact a method of Gautschi-type. Note, that this particular Gautschi-type scheme only solves problems (4.56) with $f \equiv 0$ exactly.

Remark 4.22. For $\Omega = 0$, the scheme (4.58) reduces to the leapfrog scheme applied to (4.56). ⋄

4.4.2 A low-rank version of Gautschi-type integrators

Before we develop a dynamical low-rank integrator for the full problem (4.56), we first turn our attention to the homogeneous case $f \equiv 0$ and low-rank initial values $\mathbf{A}_0, \mathbf{B}_0$ with

$$\operatorname{rank}\mathbf{A}_0 = r_{\mathbf{A}}, \qquad \operatorname{rank}\mathbf{B}_0 = r_{\mathbf{B}}$$

as usual. The solution then reads

$$\begin{bmatrix} A(t) \\ A'(t) \end{bmatrix} = \begin{bmatrix} \cos(t\Omega) & t\operatorname{sinc}(t\Omega) \\ -t\Omega^2 \operatorname{sinc}(t\Omega) & \cos(t\Omega) \end{bmatrix} \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{B}_0 \end{bmatrix} = \begin{bmatrix} \cos(t\Omega)\mathbf{A}_0 + t\operatorname{sinc}(t\Omega)\mathbf{B}_0 \\ -t\Omega^2\operatorname{sinc}(t\Omega)\mathbf{A}_0 + \cos(t\Omega)\mathbf{B}_0 \end{bmatrix}.$$

In Section 4.3 we already noted, that the ranks of $A(t)$ and $A'(t)$ are bounded by $r^* = r_{\mathbf{A}} + r_{\mathbf{B}}$. This means, if we compute low-rank approximations to $A(\tau)$ and $A'(\tau)$ employing the projector-splitting integrator with increments, cf. Algorithm 2, and approximation rank r^*, we compute matrices $\mathbf{A}_1, \mathbf{B}_1 \in \mathcal{M}_{r^*}$ satisfying $\mathbf{A}_1 = A(\tau)$ and $\mathbf{B}_1 = A'(\tau)$ due to the exactness property [Lubich and Oseledets, 2014, Theorem 4.1]. We also immediately get a low-rank factorization of $\mathbf{A}_1$ and $\mathbf{B}_1$ per construction. These then allow to perform a new step of the projector-splitting integrator to compute $\mathbf{A}_2 = A(2\tau)$ and $\mathbf{B}_2 = A'(2\tau)$ in factorized form and so on.

Remark 4.23. Since we use the approximation rank r^* in the projector-splitting integrator, we could compute the exact solution $A(t)$ in factorized form at any time by simply performing one single step of the integrator with step size $\tau = t$. As soon as $f \not\equiv 0$, this is not possible anymore. ⋄

Starting the projector-splitting integrator requires initial values $\mathbf{A}_0 = A_0$ and $\mathbf{B}_0 = B_0$ of rank r^*, i.e.,

$$\mathbf{A}_0 = \mathbf{U}_0\mathbf{S}_0\mathbf{V}_0^H, \quad \mathbf{B}_0 = \mathbf{T}_0\mathbf{R}_0\mathbf{W}_0^H, \qquad \mathbf{U}_0, \mathbf{T}_0 \in \mathcal{V}_{m,r^*}, \quad \mathbf{V}_0, \mathbf{W}_0 \in \mathcal{V}_{n,r^*}, \quad \mathbf{S}_0, \mathbf{R}_0 \in \mathbb{C}^{r^*\times r^*}.$$

Neither $\mathbf{S}_0$ nor $\mathbf{R}_0$ are invertible, since the initial values A_0 and B_0 are now over-approximated with rank r^*. The application of the projector-splitting integrator to compute the low-rank matrices $\mathbf{A}_1$ and $\mathbf{B}_1$ is nevertheless justified, since the exact solution will be of rank r^* in general and hence the over-approximation only occurs in the approximation of the initial values.

We now return to the full problem (4.56). A low-rank counterpart of the Gautschi method (4.58) is straightforwardly constructed based on its update sequences: Given approximations $\mathbf{A}_0 \approx A_0$ and $\mathbf{B}_0 \approx B_0$ of rank $r_{\mathbf{A}}$ and $r_{\mathbf{B}}$ to the initial values of (4.57), we obtain low-rank approximations $\mathbf{A}_1 \approx A(\tau)$, $\mathbf{B}_1 \approx A'(\tau)$ by first computing A_1 and B_1 as in (4.58), starting from $\mathbf{A}_0$ and $\mathbf{B}_0$. Afterwards, we get $\mathbf{A}_1$ and $\mathbf{B}_1$ as rank-$r_{\mathbf{A}}$ and rank-$r_{\mathbf{B}}$ approximations to A_1 and B_1 via the projector-splitting integrator. If we denote the numerical flow of the full-rank Gautschi method (4.58) by $\phi^{[\mathrm{G}]}$, then its low-rank equivalent can be expressed via (cf. Remark 4.1)

$$\begin{bmatrix}\mathbf{A}_1\\ \mathbf{B}_1\end{bmatrix} = \phi_\tau^{[\mathrm{G,lr}]}\begin{bmatrix}\mathbf{A}_0\\ \mathbf{B}_0\end{bmatrix} = \phi_\tau^{[\mathtt{prsi}]}\circ\phi_\tau^{[\mathrm{G}]}\begin{bmatrix}\mathbf{A}_0\\ \mathbf{B}_0\end{bmatrix}. \tag{4.61}$$

Though the low-rank version of the Gautschi method is easily derived, its efficient implementation is challenging. We postpone the presentation of the details to Chapter 5.

4.5 Approximation to A' in the tangent space

The St-LO scheme is constructed such that the approximation ranks $r_{\mathbf{A}}$ and $r_{\mathbf{B}}$ for computing low-rank approximations $\mathbf{A}$ and $\mathbf{B}$ to the exact solution A of (4.1) and its derivative A' are independent from each other. This especially means that $\mathbf{A}$ and $\mathbf{B}$ might belong to distinct low-rank manifolds.

In Corollary 2.13 it was shown, that the (time) derivative of a low-rank matrix $\mathbf{A}(t) \in \mathcal{M}_{r_{\mathbf{A}}}$ is always contained in the tangent space $\mathcal{T}_{\mathbf{A}(t)}\mathcal{M}_{r_{\mathbf{A}}}$. Sometimes, one might want to conserve this property also for the approximations $\mathbf{A}$ and $\mathbf{B}$, i.e., one is interested in approximations $\mathbf{A}_k \approx A(t_k)$, $\mathbf{B}_k \approx A'(t_k)$ satisfying

$$\mathbf{B}_k \in \mathcal{T}_{\mathbf{A}_k}\mathcal{M}_{r_{\mathbf{A}}} \qquad \text{for all} \quad k = 0, 1, \ldots. \tag{4.62}$$

We shall see in the numerical experiments, cf. Chapter 7, that the rank-adaptive variant of the new integrator (see Section 6.3) benefits from this property. I.e., there are situations where the accuracy is higher compared to the rank-adaptive variant of the St-LO scheme, while the approximation rank is smaller throughout the integration.

Remark 4.24. We already know from the representation (2.15) of elements in the tangent space to the low-rank manifold $\mathcal{M}_{r_{\mathbf{A}}}$, that the rank of such an element is bounded by $2r_{\mathbf{A}}$. However, it is not sufficient to choose $r_{\mathbf{B}} = 2r_{\mathbf{A}}$ in the St-LO scheme to conserve the tangent space relation between $\mathbf{B}$ and $\mathbf{A}$, as the manifold $\mathcal{M}_{2r_{\mathbf{A}}}$ does not coincide with the tangent space $\mathcal{T}_{\mathbf{A}}\mathcal{M}_{r_{\mathbf{A}}}$. ◇

The desired dynamical low-rank integrator for (4.1) is constructed from the non-staggered one-step formulation of the leapfrog scheme (4.3). Hence, the variant computes approximations to A and A' on the same time-grid, whereas the St-LO scheme yields approximations to A' on a staggered grid.

Let $k \in \mathbb{N}$ and let $\mathbf{A}_k \approx A(t_k)$, $\mathbf{B}_k \approx A'(t_k)$ be approximations to the exact solution to (4.1) and its derivative satisfying $\mathbf{A}_k \in \mathcal{M}_{r_\mathbf{A}}$ and $\mathbf{B}_k \in \mathcal{T}_{\mathbf{A}_k}\mathcal{M}_{r_\mathbf{A}}$, respectively. Since the rank of $\mathbf{B}_k$ is bounded from above by $2r_\mathbf{A}$, we can find a factorization of $\mathbf{B}_k$ like

$$\mathbf{B}_k = \mathbf{T}_k \mathbf{R}_k \mathbf{W}_k^H, \qquad \mathbf{T}_k \in \mathcal{V}_{m,2r_\mathbf{A}}, \quad \mathbf{W}_k \in \mathcal{V}_{n,2r_\mathbf{A}}, \quad \mathbf{R}_k \in \mathbb{C}^{2r_\mathbf{A} \times 2r_\mathbf{A}}, \tag{4.63}$$

where $\mathbf{R}_k$ might by singular. We now compute approximations $\mathbf{A}_{k+1} \in \mathcal{M}_{r_\mathbf{A}}$ and $\mathbf{B}_{k+1} \in \mathcal{T}_{\mathbf{A}_k}\mathcal{M}_{r_\mathbf{A}}$ by the following update steps:

1. Compute a rank-$2r_\mathbf{A}$ approximation to $A'(t_k + \frac{\tau}{2})$ by approximately solving
$$\widetilde{B}'(\sigma) = F(\mathbf{A}_k), \quad \sigma \in [0, \frac{\tau}{2}], \qquad \widetilde{B}(0) = \mathbf{B}_k,$$
using the projector-splitting integrator. This yields an approximation $\mathbf{B}_{k+\frac{1}{2}} = \mathbf{T}_{k+\frac{1}{2}} \mathbf{R}_{k+\frac{1}{2}} \mathbf{W}_{k+\frac{1}{2}}^H \in \mathcal{M}_{2r_\mathbf{A}}$.

2. Compute a rank-$r_\mathbf{A}$ approximation to $A(t_k + \tau)$ by approximately solving
$$\widetilde{A}'(\sigma) = \mathbf{B}_{k+\frac{1}{2}}, \quad \sigma \in [0, \tau], \qquad \widetilde{A}(0) = \mathbf{A}_k,$$
using the projector-splitting integrator, yielding $\mathbf{A}_{k+1} = \mathbf{U}_{k+1}\mathbf{S}_{k+1}\mathbf{V}_{k+1}^H \in \mathcal{M}_{r_\mathbf{A}}$.

3. Finally, compute
$$\widetilde{B}_{k+1} = \mathbf{B}_{k+\frac{1}{2}} + \frac{\tau}{2} F(\mathbf{A}_{k+1}),$$
and obtain $\mathbf{B}_{k+1}$ by projection,
$$\mathbf{B}_{k+1} = \widetilde{\Pi}_{\mathbf{A}_{k+1}} \widetilde{B}_{k+1}.$$
This indeed yields $\mathbf{B}_{k+1} \in \mathcal{T}_{\mathbf{A}_{k+1}}\mathcal{M}_{r_\mathbf{A}}$.

For an efficient scheme, it is favorable if this update sequence naturally yields a low-rank decomposition of $\mathbf{B}_{k+1}$ similar to (4.63). Furthermore, computing $\widetilde{B}_{k+1}$ explicitly, i.e., its full-dimensional form, and performing the projection afterwards neglects the benefits admitted by the low-rank factorization which usually is available for dynamical low-rank schemes. Therefore the third substep needs to be handled carefully in order to maintain efficiency.

We first consider a simpler, but related situation: For given $Z \in \mathbb{C}^{m \times n}$ and $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H \in \mathcal{M}_{r_\mathbf{A}}$, we want to compute a low-rank decomposition of $\widetilde{\Pi}_\mathbf{A} Z = \mathbf{T}\mathbf{R}\mathbf{W}^H \in \mathcal{T}_\mathbf{A}\mathcal{M}_{r_\mathbf{A}}$, where the factors satisfy $\mathbf{T} \in \mathcal{V}_{m,2r_\mathbf{A}}, \mathbf{W} \in \mathcal{V}_{n,2r_\mathbf{A}}$, and $\mathbf{R} \in \mathbb{C}^{2r_\mathbf{A} \times 2r_\mathbf{A}}$ as in (4.63). From the representation (2.21) of the orthogonal projector onto $\mathcal{T}_\mathbf{A}\mathcal{M}_{r_\mathbf{A}}$, we get

$$\begin{aligned}
\widetilde{\Pi}_\mathbf{A} Z &= Z\Pi_\mathbf{V} - \Pi_\mathbf{U} Z \Pi_\mathbf{V} + \Pi_\mathbf{U} Z \\
&= \Pi_\mathbf{U} Z \Pi_\mathbf{V} + \Pi_\mathbf{U}^\perp Z \Pi_\mathbf{V} + \Pi_\mathbf{U} Z \Pi_\mathbf{V}^\perp \\
&= \mathbf{U}(\mathbf{U}^H Z \mathbf{V})\mathbf{V}^H + (\Pi_\mathbf{U}^\perp Z \mathbf{V})\mathbf{V}^H + \mathbf{U}(\mathbf{U}^H Z \Pi_\mathbf{V}^\perp).
\end{aligned} \tag{4.64}$$

We now set

$$\mathbf{M} = \mathbf{U}^H Z \mathbf{V} \in \mathbb{C}^{r_\mathbf{A} \times r_\mathbf{A}}, \qquad \widetilde{\mathbf{U}} = (\Pi_\mathbf{U}^\perp Z \mathbf{V}) \in \mathbb{C}^{m \times r_\mathbf{A}}, \qquad \widetilde{\mathbf{V}} = (\mathbf{U}^H Z \Pi_\mathbf{V}^\perp)^H \in \mathbb{C}^{n \times r_\mathbf{A}},$$

where $\mathbf{U}^H\widetilde{\mathbf{U}} = 0$ and $\mathbf{V}^H\widetilde{\mathbf{V}} = 0$. Thus (4.64) matches the representation (2.15) of elements in the tangent space $\mathcal{T}_{\mathbf{A}}\mathcal{M}_{r_{\mathbf{A}}}$, and we directly obtain the factorization

$$\widetilde{\Pi}_{\mathbf{A}}Z = \widetilde{\mathbf{T}}\widetilde{\mathbf{R}}\widetilde{\mathbf{W}}^H = \begin{bmatrix}\mathbf{U} & \widetilde{\mathbf{U}}\end{bmatrix}\begin{bmatrix}\mathbf{M} & I_{r_{\mathbf{A}}}\\ I_{r_{\mathbf{A}}} & 0\end{bmatrix}\begin{bmatrix}\mathbf{V} & \widetilde{\mathbf{V}}\end{bmatrix}^H.$$

However, the factors $\widetilde{\mathbf{T}}$ and $\widetilde{\mathbf{W}}$ are not elements of the Stiefel-manifolds $\mathcal{V}_{m,2r_{\mathbf{A}}}$ and $\mathcal{V}_{n,2r_{\mathbf{A}}}$, respectively, since in general neither $\widetilde{\mathbf{U}}$ nor $\widetilde{\mathbf{V}}$ are orthonormal. We therefore compute subsequent QR-decompositions

$$\mathbf{T}\mathbf{R}_{\mathbf{T}} = \widetilde{\mathbf{T}}, \quad \mathbf{T} \in \mathcal{V}_{m,2r_{\mathbf{A}}}, \qquad \mathbf{W}\mathbf{R}_{\mathbf{W}} = \widetilde{\mathbf{W}}, \quad \mathbf{W} \in \mathcal{V}_{n,2r_{\mathbf{A}}}.$$

Setting $\mathbf{R} = \mathbf{R}_{\mathbf{T}}\widetilde{\mathbf{R}}\mathbf{R}_{\mathbf{W}}^H$ finally yields the desired factorization.

Remark 4.25. If the matrix $\widetilde{\mathbf{U}}$ has full rank $r_{\mathbf{A}}$, it is not necessary to compute the QR-decomposition of $\widetilde{\mathbf{T}}$ but it suffices to decompose $\widetilde{\mathbf{U}}$ instead: Denote by $\mathbf{Q}_{\widetilde{\mathbf{U}}}\mathbf{R}_{\widetilde{\mathbf{U}}} = \widetilde{\mathbf{U}}$ the QR-decomposition of $\widetilde{\mathbf{U}}$. By assumption, $\mathbf{R}_{\widetilde{\mathbf{U}}}$ is invertible and hence

$$\mathbf{U}^H\widetilde{\mathbf{U}} = \mathbf{U}^H\mathbf{Q}_{\widetilde{\mathbf{U}}}\mathbf{R}_{\widetilde{\mathbf{U}}} = 0 \quad\Longrightarrow\quad \mathbf{U}^H\mathbf{Q}_{\widetilde{\mathbf{U}}} = 0, \qquad \text{where} \quad \mathbf{U} \in \mathcal{V}_{m,r_{\mathbf{A}}}, \quad \mathbf{Q}_{\widetilde{\mathbf{U}}} \in \mathcal{V}_{m,r_{\mathbf{A}}}.$$

Then $\widetilde{\mathbf{T}}$ admits the decomposition

$$\widetilde{\mathbf{T}} = \mathbf{T}\mathbf{R}_{\mathbf{T}} = \begin{bmatrix}\mathbf{U} & \mathbf{Q}_{\widetilde{\mathbf{U}}}\end{bmatrix}\begin{bmatrix}I_{r_{\mathbf{A}}} & 0\\ 0 & \mathbf{R}_{\widetilde{\mathbf{U}}}\end{bmatrix}, \qquad \mathbf{T} \in \mathcal{V}_{m,2r_{\mathbf{A}}}.$$

Likewise, if $\widetilde{\mathbf{V}}$ has full rank $r_{\mathbf{A}}$, with the same argument one shows the decomposition

$$\widetilde{\mathbf{W}} = \mathbf{W}\mathbf{R}_{\mathbf{W}} = \begin{bmatrix}\mathbf{V} & \mathbf{Q}_{\widetilde{\mathbf{V}}}\end{bmatrix}\begin{bmatrix}I_{r_{\mathbf{A}}} & 0\\ 0 & \mathbf{R}_{\widetilde{\mathbf{V}}}\end{bmatrix},$$

where $\mathbf{Q}_{\widetilde{\mathbf{V}}}\mathbf{R}_{\widetilde{\mathbf{V}}} = \widetilde{\mathbf{V}}$ is the QR-decomposition of $\widetilde{\mathbf{V}}$ and $\mathbf{W} \in \mathcal{V}_{n,2r_{\mathbf{A}}}$. ◇

Overall, the efficient implementation of the third substep above hence depends on the efficient computation of the matrices $\mathbf{M}, \widetilde{\mathbf{U}}$, and $\widetilde{\mathbf{V}}$. In our particular case we have $Z = \widetilde{B}_{k+1} = \mathbf{B}_{k+\frac{1}{2}} + \frac{\tau}{2}F(\mathbf{A}_{k+1})$, which allows us to compute these by clever ordering of the matrix-products: Define

$$\begin{aligned}\mathbf{Y} &= Z\mathbf{V}_{k+1}\\ &= (\mathbf{T}_{k+\frac{1}{2}}\mathbf{R}_{k+\frac{1}{2}}\mathbf{W}_{k+\frac{1}{2}}^H + \frac{\tau}{2}F(\mathbf{U}_{k+1}\mathbf{S}_{k+1}\mathbf{V}_{k+1}^H))\mathbf{V}_{k+1}\\ &= (\mathbf{T}_{k+\frac{1}{2}}\mathbf{R}_{k+\frac{1}{2}})(\mathbf{W}_{k+\frac{1}{2}}^H\mathbf{V}_{k+1}) + \frac{\tau}{2}F(\mathbf{U}_{k+1}\mathbf{S}_{k+1}\mathbf{V}_{k+1}^H)\mathbf{V}_{k+1}\\ &= (\mathbf{T}_{k+\frac{1}{2}}\mathbf{R}_{k+\frac{1}{2}})\mathbf{X} + \frac{\tau}{2}\mathbf{K},\end{aligned}$$

where

$$\mathbf{X} = \mathbf{W}_{k+\frac{1}{2}}^H\mathbf{V}_{k+1}, \qquad \mathbf{K} = F(\mathbf{U}_{k+1}\mathbf{S}_{k+1}\mathbf{V}_{k+1}^H)\mathbf{V}_{k+1}.$$

Then it is

$$\mathbf{M} = \mathbf{U}_{k+1}^H\mathbf{Y}, \qquad \text{as well as} \qquad \widetilde{\mathbf{U}} = \mathbf{Y} - \mathbf{U}_{k+1}\mathbf{M}.$$

With

$$\mathbf{L} = F(\mathbf{U}_{k+1}\mathbf{S}_{k+1}\mathbf{V}_{k+1}^H)^H\mathbf{U}_{k+1}, \qquad \mathbf{X} = \mathbf{T}_{k+\frac{1}{2}}^H\mathbf{U}_{k+1},$$

and

$$\begin{aligned}\widetilde{\mathbf{Y}} &= Z^H \mathbf{U}_{k+1} \\ &= \mathbf{W}_{k+\frac{1}{2}} \mathbf{R}_{k+\frac{1}{2}}^H \mathbf{T}_{k+\frac{1}{2}}^H \mathbf{U}_{k+1} + \frac{\tau}{2} F(\mathbf{U}_{k+1}\mathbf{S}_{k+1}\mathbf{V}_{k+1}^H)^H \mathbf{U}_{k+1} \\ &= \mathbf{W}_{k+\frac{1}{2}} \mathbf{R}_{k+\frac{1}{2}}^H \mathbf{X} + \frac{\tau}{2}\mathbf{L},\end{aligned}$$

one then obtains

$$\widetilde{\mathbf{V}} = \widetilde{\mathbf{Y}} - \mathbf{V}_{k+1}\mathbf{M}^H,$$

see also Algorithm 9. We thus make use of the available factorization of $\mathbf{B}_{k+\frac{1}{2}}$ as well as the efficient routines for computing products $F(\mathbf{A})\mathbf{V}$ and $F(\mathbf{A})^H\mathbf{U}$ like mentioned in Remark 4.2. A single time step of the overall scheme for the integration of (4.1) with property (4.62) is presented in Algorithm 10.

Algorithm 9: Projection step according to Section 4.5 for usage in Algorithm 10

```
1  projection($\mathbf{U}, \mathbf{S}, \mathbf{V}, \mathbf{T}, \mathbf{R}, \mathbf{W}, r_\mathbf{A}, \tau, F$)
   Input  : factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H \in \mathcal{M}_{r_\mathbf{A}}$ satisfying (4.9a), factors $\mathbf{T}, \mathbf{R}, \mathbf{W}$ of
            $\mathbf{B} = \mathbf{T}\mathbf{R}\mathbf{W}^H \in \mathcal{M}_{2r_\mathbf{A}}$ satisfying (4.9b) with $r_\mathbf{B} = 2r_\mathbf{A}$, step size $\tau$, right-hand side $F$
2  $\mathbf{K} = F(\mathbf{U}\mathbf{S}\mathbf{V}^H)\mathbf{V}$
3  $\mathbf{X} = \mathbf{W}^H\mathbf{V}$
4  $\mathbf{Y} = \mathbf{T}\mathbf{R}\mathbf{X} + \frac{\tau}{2}\mathbf{K}$
5  $\mathbf{M} = \mathbf{U}^H\mathbf{Y}$
6  $\widetilde{\mathbf{U}} = \mathbf{Y} - \mathbf{U}\mathbf{M}$
7  $\mathbf{L} = F(\mathbf{U}\mathbf{S}\mathbf{V}^H)^H\mathbf{U}$
8  $\mathbf{X} = \mathbf{T}^H\mathbf{U}$
9  $\widetilde{\mathbf{Y}} = \mathbf{W}\mathbf{R}^H\mathbf{X} + \frac{\tau}{2}\mathbf{L}$
10 $\widetilde{\mathbf{V}} = \widetilde{\mathbf{Y}} - \mathbf{V}\mathbf{M}^H$
11 $\widetilde{\mathbf{T}}_1 = \begin{bmatrix}\mathbf{U} & \widetilde{\mathbf{U}}\end{bmatrix}, \quad \widetilde{\mathbf{W}}_1 = \begin{bmatrix}\mathbf{V} & \widetilde{\mathbf{V}}\end{bmatrix}, \quad \widetilde{\mathbf{R}}_1 = \begin{bmatrix}\mathbf{M} & I_{r_\mathbf{A}} \\ I_{r_\mathbf{A}} & 0\end{bmatrix}$
12 Compute reduced $QR$-decompositions $\mathbf{T}_1\mathbf{R}_\mathbf{T} = \widetilde{\mathbf{T}}_1$ and $\mathbf{W}_1\mathbf{R}_\mathbf{W} = \widetilde{\mathbf{W}}_1$
13 Set $\mathbf{R}_1 = \mathbf{R}_\mathbf{T}\widetilde{\mathbf{R}}_1\mathbf{R}_\mathbf{W}^H$
14 Return $\mathbf{T}_1, \mathbf{R}_1, \mathbf{W}_1$
   Output: factors $\mathbf{T}_1, \mathbf{R}_1, \mathbf{W}_1$ of $\mathbf{B}_1 = \widetilde{\Pi}_\mathbf{A}\big(\mathbf{B} + \frac{\tau}{2}F(\mathbf{A})\big) \in \mathcal{T}_\mathbf{A}\mathcal{M}_{r_\mathbf{A}}$ satisfying (4.63)
```

Algorithm 10: Dynamical low-rank integrator for second-order problems (4.1) with approximations satisfying (4.62), single time step

```
1 stlotangent($\mathbf{U},\mathbf{S},\mathbf{V},\mathbf{T},\mathbf{R},\mathbf{W},r_{\mathbf{A}},\tau,F$)
  Input  : factors $\mathbf{U},\mathbf{S},\mathbf{V}$ of rank-$r_{\mathbf{A}}$ approximation $\mathbf{A}=\mathbf{U}\mathbf{S}\mathbf{V}^H \approx A(t)$, factors $\mathbf{T},\mathbf{R},\mathbf{W}$ of
           approximation $\mathbf{B}\in\mathcal{T}_{\mathbf{A}}\mathcal{M}_{r_{\mathbf{A}}}$ satisfying (4.63), $\mathbf{B}=\mathbf{T}\mathbf{R}\mathbf{W}^H\in\mathcal{T}_{\mathbf{A}}\mathcal{M}_{r_{\mathbf{A}}}$, step size $\tau$,
           right-hand side $F$
2 B-step: $\mathbf{T}_{\frac{1}{2}},\mathbf{R}_{\frac{1}{2}},\mathbf{W}_{\frac{1}{2}},\mathbf{L}=$ prsi$\big(\mathbf{T},\mathbf{R},\mathbf{W},r_{\mathbf{B}},\Delta B\big)$   where $\Delta B=\frac{\tau}{2}F(\mathbf{U}\mathbf{S}\mathbf{V}^H)$
3 A-step:  $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1,\mathbf{L}_1=$ prsi$\big(\mathbf{U},\mathbf{S},\mathbf{V},r_{\mathbf{A}},\Delta A\big)$   where $\Delta A=\tau\mathbf{T}_1\mathbf{L}^H$
4 B-step:    $\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1=$ projection$\big(\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1,\mathbf{T}_{\frac{1}{2}},\mathbf{R}_{\frac{1}{2}},\mathbf{W}_{\frac{1}{2}},\tau,F\big)$
5 Return $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1,\mathbf{L}_1,\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1$
  Output: factors $\mathbf{U}_1,\mathbf{S}_1,\mathbf{V}_1$ of rank-$r_{\mathbf{A}}$ approximation $\mathbf{A}_1=\mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H\approx A(t+\tau)$ satisfying
          (4.9a), $\mathbf{L}_1=\mathbf{V}_1\mathbf{S}_1^H$, factors $\mathbf{T}_1,\mathbf{R}_1,\mathbf{W}_1$ of approximation $\mathbf{B}_1\in\mathcal{T}_{\mathbf{A}_1}\mathcal{M}_{r_{\mathbf{A}}}$ satisfying
          (4.63), $\mathbf{B}_1=\mathbf{T}_1\mathbf{R}_1\mathbf{W}_1\approx A'(t+\tau)$
```

CHAPTER 5

Implementation of dynamical low-rank integrators

In the previous chapters we revised and introduced several dynamical low-rank integrators for first- and second-order matrix differential equations. As already mentioned, all these integrators, despite their differences, share one fundamental difficulty, namely their efficient implementation. For us, *efficient implementation* is twofold: On the one hand, we aim for a fast computation, i.e., we want the methods to yield low-rank approximations in a short amount of time. On the other hand, we also want an efficient storage management. In particular we aim for an implementation of the schemes that avoids computations with matrices of size $m \times n$. This is motivated by the fact that all dynamical low-rank integrators yield approximations in factorized form, with factors of r columns, where $r \in \mathbb{N}$ is the approximation rank. Also, all update steps of all dynamical low-rank integrators are performed such that only matrices of column dimension r are updated. Recall that for any dynamical low-rank integrator discussed within this thesis, the right-hand side F always appears in a matrix product with a matrix $\mathbf{E}$ of column dimension r. Computing the product $F(\mathbf{A})\mathbf{E}$ based on the low-rank decomposition of $\mathbf{A}$ is crucial for all dynamical low-rank integrators. This is discussed in detail in Section 5.1.

Furthermore, we comment on the situation where we have matrix functions (cf. Definition A.20) involved in the dynamical low-rank schemes. This is the case for the `prsistiff` (cf. Section 3.3), the `stlostiff` (cf. Section 4.3), and the `gautschilr` (cf. Section 4.4.2) schemes. The calculation of, e.g., the matrix exponential usually suffers from long computation times if the dimension of the argument is large. In Section 5.2 we focus on two solutions of this problem: Implementation by diagonalization of the argument, e.g., by FFT routines, and iterative methods.

In Section 5.3, we proceed with the implementation of the `gautschilr` method postponed from Section 4.4.2.

5.1 Implementation of low-rank matrix products for dynamical low-rank integrators

In this section, we comment on the implementation of the products

$$F(\mathbf{A})\mathbf{E}, \qquad \mathbf{E} \in \mathbb{C}^{n\times r}, \tag{5.1}$$

for $\mathbf{A}$ given by its low-rank factorization $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H \in \mathcal{M}_r$. Here F is the right-hand side of the first-order matrix differential equation (3.1) or the second-order matrix differential equation (4.1), respectively, and $\mathbf{E}$ is an arbitrary slim matrix. The aim is to avoid to compute the mn entries of $\mathbf{A}$ explicitly. In most cases, this requires a non-intuitive computation of the products in (5.1). The implementation of products $F(\mathbf{A})^H\mathbf{E}$ for $\mathbf{E} \in \mathbb{C}^{m\times r}$ is realized analogously.

As presented in Chapter 3, the projector-splitting integrator does not only yield the factors $\mathbf{U}$, $\mathbf{S}$, and $\mathbf{V}$ of $\mathbf{A}$, but also the product $\mathbf{L} = \mathbf{V}\mathbf{S}^H$ at no additional cost. As a consequence, we present the implementation of the products in (5.1) starting from the decomposition

$$\mathbf{A} = \mathbf{U}\mathbf{L}^H, \qquad \mathbf{U} \in \mathcal{V}_{m,r}, \quad \mathbf{L} \in \mathbb{C}^{n\times r}. \tag{5.2}$$

Remark 5.1. One could exchange the projector-splitting integrator by the unconventional robust integrator, cf. Section 3.4.2, in all dynamical low-rank integrators for first-order and second-order matrix differential equations, respectively. Then the matrix $\mathbf{L} = \mathbf{V}\mathbf{S}^H$ is not provided by the algorithms. However, if r is small compared to n, the computation of the product $\mathbf{L}$ from the factors $\mathbf{V}$ and $\mathbf{S}$ is cheap. ◇

In the following, we first consider linear functions F in (5.1). Afterwards, we focus on a general approach for computing products of type (5.1) when the right-hand side is evaluated *entrywise*. As a special case, we consider monomials. Any linear combination of these types of right-hand sides can be implemented by combining the presented strategies.

5.1.1 Linear right-hand sides

Consider $F : \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ given by

$$F(\mathbf{A}) = L_1\mathbf{A} + \mathbf{A}L_2,$$

where $L_1 \in \mathbb{C}^{m\times m}$ and $L_2 \in \mathbb{C}^{n\times n}$. We have

$$\begin{aligned} F(\mathbf{A})\mathbf{E} &= (L_1\mathbf{A} + \mathbf{A}L_2)\mathbf{E} \\ &= L_1\mathbf{U}\mathbf{L}^H\mathbf{E} + \mathbf{U}\mathbf{L}^H L_2\mathbf{E} \\ &= L_1\big(\mathbf{U}(\mathbf{L}^H\mathbf{E})\big) + \mathbf{U}\big(\mathbf{L}^H(L_2\mathbf{E})\big). \end{aligned} \tag{5.3}$$

Computing from the inner parentheses to the outer ones ensures that no matrix with more than r columns is involved, and hence yields a storage-economical implementation.

We point out that the above approach generalizes straightforwardly to the situation, where L_1 or L_2 are given in a factorized form, e.g., in a decomposition like

$$L_1 = Q^H\Omega_1^2 Q, \qquad Q \in \mathbb{C}^{m\times m} \text{ orthonormal.}$$

5.1.2 Nonlinear entrywise functions

Consider now the product

$$\mathbf{C} = f(\mathbf{A})\mathbf{E}, \qquad \mathbf{E} \in \mathbb{C}^{n\times r} \text{ arbitrary}, \tag{5.4}$$

where $f : \mathbb{C} \to \mathbb{C}$ and

$$f(\mathbf{A})_{ij} = f(\mathbf{A}_{ij}) \qquad \text{for all } i = 1, \dots, m,\ j = 1, \dots, n.$$

Based on the representation (5.2) of $\mathbf{A} \in \mathcal{M}_r$, we can compute the product (5.4) storage-economical.

Denote the ith standard basis vector of $\mathbb{R}^m$ as e_i. Since f is entrywise evaluated, the first column of $\mathbf{C}$ can be expressed via

$$e_i^T\mathbf{C} = f(e_i^T\mathbf{A})\mathbf{E} \in \mathbb{C}^{1\times r}, \qquad i = 1, \dots, m.$$

$\mathbf{A}$ is available in factorized form, so that we can compute the ith row of $\mathbf{A}$ by the slim matrices $\mathbf{U}$ and $\mathbf{L}$ via

$$e_i^T\mathbf{A} = (\mathrm{e}_i^T\,\mathbf{U})\mathbf{L}^H \in \mathbb{C}^{1\times n}, \qquad i = 1, \dots, m.$$

The products are computed quickly if r is small. This sequence of operations can be carried out for every row index i independently, and thus the computation of the rows of the solution $\mathbf{C} = f(\mathbf{A})\mathbf{E}$ can be done in parallel.

This general approach for computing the low-rank products (5.4) can be used for all entrywise evaluated functions f, including polynomials of arbitrary degree, trigonometric functions, exponential or logarithmic functions, and any combination of these. However, especially when the rank r is very small compared to the dimensions m and n of $\mathbf{A} \in \mathcal{M}_r$, there are situations where one can achieve a slightly faster computation. Such alternatives are often tailored to the respective right-hand side f and thus cannot be generalized. One specific example we nevertheless want to discuss are entrywise power functions. This is done in the next section.

5.1.3 Power functions

We here consider the special choice $f(z) = z^p\overline{z}^q$, $p, q \in \mathbb{N}$, $p+q \geq 2$ for $z \in \mathbb{C}$, where $\overline{z}$ denotes the complex conjugate of z. The entrywise evaluation of f can be written in terms of the Hadamard product $\bullet$,

$$f(\mathbf{A}) = \underbrace{\mathbf{A} \bullet \mathbf{A} \bullet \dots \bullet \mathbf{A}}_{p \text{ times}} \bullet \underbrace{\overline{\mathbf{A} \bullet \mathbf{A} \bullet \dots \bullet \mathbf{A}}}_{q \text{ times}}.$$

We denote the jth column of a matrix $A \in \mathbb{C}^{m\times n}$ by A_j. The matrix $\mathbf{A} = \mathbf{U}\mathbf{L}^H \in \mathcal{M}_r$ as in (5.2) can then be expressed as the following sum of rank 1 matrices,

$$\mathbf{A} = \mathbf{U}\mathbf{L}^H = \sum_{j=1}^{r} \mathbf{U}_j\mathbf{L}_j^H. \tag{5.5}$$

The Hadamard product is distributive and satisfies

$$(\mathbf{U}_j\mathbf{L}_j^H) \bullet (\mathbf{U}_k\mathbf{L}_k^H) = (\mathbf{U}_j \bullet \mathbf{U}_k)(\mathbf{L}_j \bullet \mathbf{L}_k)^H, \qquad 1 \leq j, k \leq r,$$

cf. [Styan, 1973, Section 2]. Hence, for $\mathbf{E} \in \mathbb{C}^{n\times r}$ it holds

$$\begin{aligned}
f(\mathbf{A})\mathbf{E} &= \left(\Big(\sum_{j_1=1}^{r} \mathbf{U}_{j_1}\mathbf{L}_{j_1}^H\Big) \bullet \ldots \bullet \Big(\sum_{j_p=1}^{r} \mathbf{U}_{j_p}\mathbf{L}_{j_p}^H\Big) \bullet \overline{\Big(\sum_{k_1=1}^{r} \mathbf{U}_{k_1}\mathbf{L}_{k_1}^H\Big) \bullet \ldots \bullet \Big(\sum_{k_q=1}^{r} \mathbf{U}_{k_q}\mathbf{L}_{k_q}^H\Big)}\right)\mathbf{E} \\
&= \left(\sum_{\substack{j_1,\ldots,j_p,\\ k_1,\ldots,k_q=1}}^{r} (\mathbf{U}_{j_1} \bullet \ldots \bullet \mathbf{U}_{j_p} \bullet \overline{\mathbf{U}}_{k_1} \bullet \ldots \bullet \overline{\mathbf{U}}_{k_q})(\mathbf{L}_{j_1} \bullet \ldots \bullet \mathbf{L}_{j_p} \bullet \overline{\mathbf{L}}_{k_1} \bullet \ldots \bullet \overline{\mathbf{L}}_{k_q})^H\right)\mathbf{E} \\
&= \sum_{\substack{j_1,\ldots,j_p,\\ k_1,\ldots,k_q=1}}^{r} (\mathbf{U}_{j_1} \bullet \ldots \bullet \mathbf{U}_{j_p} \bullet \overline{\mathbf{U}}_{k_1} \bullet \ldots \bullet \overline{\mathbf{U}}_{k_q})\Big((\mathbf{L}_{j_1} \bullet \ldots \bullet \mathbf{L}_{j_p} \bullet \overline{\mathbf{L}}_{k_1} \bullet \ldots \bullet \overline{\mathbf{L}}_{k_q})^H\mathbf{E}\Big).
\end{aligned}$$

We now consider the special case of the cubic nonlinearity

$$f(\mathbf{A}) = \mathbf{A} \bullet \overline{\mathbf{A}} \bullet \mathbf{A}.$$

We then have

$$f(\mathbf{A})\mathbf{E} = \sum_{j,k,\ell=1}^{r} \mathbf{U}_{jk\ell}\mathbf{L}_{jk\ell}^H, \qquad \text{where} \quad \mathbf{U}_{jk\ell} = \mathbf{U}_j \bullet \overline{\mathbf{U}}_k \bullet \mathbf{U}_\ell, \qquad \mathbf{L}_{jk\ell} = \mathbf{L}_j \bullet \overline{\mathbf{L}}_k \bullet \mathbf{L}_\ell.$$

Due to the symmetry of $\mathbf{U}_{jk\ell}$ and $\mathbf{L}_{jk\ell}$ in j and ℓ, respectively, this can be rewritten as

$$f(\mathbf{A})\mathbf{E} = \sum_{k=1}^{r}\Big[\sum_{j=1}^{r}(\mathbf{U}_j^2 \bullet \overline{\mathbf{U}}_k)\big((\mathbf{L}_j^2 \bullet \overline{\mathbf{L}}_k)^H\mathbf{E}\big) + 2\sum_{j=1}^{r}\sum_{\ell=1}^{j-1}\mathbf{U}_{jk\ell}(\mathbf{L}_{jk\ell}^H\mathbf{E})\Big]. \tag{5.6}$$

Here one avoids computing the same matrices multiple times, which speeds up the computation.

Lastly, with regard to Section 7.2.2 below, we consider

$$f(\mathbf{A}) = \boldsymbol{\chi} \bullet \mathbf{A} \bullet \overline{\mathbf{A}} \bullet \mathbf{A},$$

where

$$\boldsymbol{\chi}_{ij} = \begin{cases} 1, & \eta \le i \le \xi \quad \text{and} \quad \mu \le j \le \nu, \\ 0, & \text{else,} \end{cases}$$

and η, ξ, μ, and ν are fixed integer values satisfying

$$1 \le \eta \le \xi \le m, \qquad 1 \le \mu \le \nu \le n.$$

The matrix $\boldsymbol{\chi}$ admits the alternative representation

$$\boldsymbol{\chi} = \begin{pmatrix} \mathbb{O}_{\eta-1} \\ \mathbb{1}_{\xi-\eta+1} \\ \mathbb{O}_{m-\xi} \end{pmatrix} \begin{pmatrix} \mathbb{O}_{\mu-1} \\ \mathbb{1}_{\nu-\mu+1} \\ \mathbb{O}_{n-\nu} \end{pmatrix}^T =: \widetilde{\mathbb{1}}_m \widetilde{\mathbb{1}}_n.$$

Here, $\mathbb{O}_{m\times n}$ and $\mathbb{1}_{m\times n}$ are the matrices of dimension $m \times n$ with all entries being zeros and ones, respectively, and $\mathbb{O}_m = \mathbb{O}_{m\times 1}$ and $\mathbb{1}_m = \mathbb{1}_{m\times 1}$. Hence, $\boldsymbol{\chi}$ can be seen as a restriction matrix. In the following, we denote the restriction of some vector $x \in \mathbb{C}^m$ to its ηth to ξth entries by $\vartheta(x) \in \mathbb{C}^{\xi-\eta+1}$, and the restriction of some vector $y \in \mathbb{C}^{1\times n} \cong \mathbb{C}^n$ to its μth to νth entries by $\varrho(y) \in \mathbb{C}^{\nu-\mu+1}$. Since for the Hadamard product it holds for any $A \in \mathbb{C}^{m\times n}$, $x \in \mathbb{R}^m$, and $y \in \mathbb{R}^n$, cf. [Styan, 1973, Section 2],

$$(xy^T) \bullet A = \operatorname{diag}(x) A \operatorname{diag}(y),$$

we have from (5.5)

$$\begin{aligned}(\boldsymbol{\chi} \bullet \mathbf{A})\mathbf{E} &= \big[(\widetilde{\mathbb{1}}_m \widetilde{\mathbb{1}}_n) \bullet \mathbf{A}\big]\mathbf{E} \\ &= \Big[\operatorname{diag}(\widetilde{\mathbb{1}}_m)\mathbf{A}\operatorname{diag}(\widetilde{\mathbb{1}}_n)\Big]\mathbf{E} \\ &= \left[\operatorname{diag}(\widetilde{\mathbb{1}}_m)\Big(\sum_{j=1}^{r}\mathbf{U}_j\mathbf{L}_j^H\Big)\operatorname{diag}(\widetilde{\mathbb{1}}_n)\right]\mathbf{E} \\ &= \Big[\sum_{j=1}^{r}\big(\operatorname{diag}(\widetilde{\mathbb{1}}_m)\mathbf{U}_j\big)\mathbf{L}_j^H\Big]\big(\operatorname{diag}(\widetilde{\mathbb{1}}_n)\mathbf{E}\big) \\ &= \begin{bmatrix} \mathbb{0}_{(\eta-1)\times r} \\ \sum_{j=1}^{r}\vartheta(\mathbf{U}_j)\varrho(\mathbf{L}_j)^H\varrho(\mathbf{E}) \\ \mathbb{0}_{(m-\xi)\times r}\end{bmatrix}. \end{aligned} \tag{5.7}$$

Hence, it suffices to compute the small matrices $\vartheta(\mathbf{U}_j)\varrho(\mathbf{L}_j)^H\varrho(\mathbf{E})$ and summing them up. The implementation of $(\boldsymbol{\chi} \bullet \mathbf{A} \bullet \overline{\mathbf{A}} \bullet \mathbf{A})\mathbf{E}$ is realized by combining (5.6) and (5.7).

5.2 Computation of matrix functions

As seen in Sections 3.3, 4.3, and 4.4.2, the constructed dynamical low-rank integrators tailored to stiff matrix differential equations and oscillatory problems contain matrix functions. Thus, for a matrix $L \in \mathbb{C}^{m\times m}$ we are interested in the computation of $g(\tau L)\mathbf{E}$, where $\mathbf{E} \in \mathbb{C}^{n\times r}$, $\tau > 0$, and g is an arbitrary analytic function. For time integration, the most relevant is the matrix exponential $\mathrm{e}^{\tau L}$, which is present in the `prsistiff` integrator introduced in Section 3.3. Further examples are

$$\cos(\tau L), \qquad \operatorname{sinc}(\tau L), \qquad L^2\operatorname{sinc}(\tau L), \tag{5.8}$$

but also the filter functions (4.59) arising in Gautschi-type methods.

5.2.1 Implementation by diagonalization

Assume that $L \in \mathbb{C}^{m\times m}$ admits the decomposition

$$L = Q^{-1}\Lambda Q, \qquad \Lambda = \operatorname{diag}(\lambda_i) \in \mathbb{C}^{m\times m} \text{ diagonal}, \qquad Q \in \mathbb{C}^{m\times m} \text{ invertible}.$$

By Theorem A.21 it is

$$g(\tau L) = Q^{-1}g(\tau\Lambda)Q, \tag{5.9}$$

and the argument of the matrix function is now diagonal, namely

$$g(\tau\Lambda) = \operatorname{diag}\big(g(\tau\lambda_i)\big).$$

If Q and Q^{-1} are known, the computation of $g(\tau L)\mathbf{E}$ can be realized by successive matrix multiplication similarly to (5.3). Together with (5.9) all matrix functions in (5.8) can be computed exactly, up to round-off errors.

While the computation of $g(\tau L)\mathbf{E}$ by diagonalization seems favorable, it might suffer from great effort. Often L is sparse, which is common if L originates from the space discretization of a first or second-order partial differential operator. Unfortunately, the matrix Q is in general dense, and the multiplication of Q with a vector requires $\mathcal{O}(m^2)$ operations. We therefore use this ansatz when the multiplication with Q is of reasonable computational effort, e.g., if $Q = \mathcal{F}_m$ is the discrete Fourier matrix. Then, a multiplication with $\mathcal{F}_m$ takes $\mathcal{O}(m \log m)$ operations as soon as FFT routines are employed.

Remark 5.2. Due to its relation to the discrete Fourier transform, we also compute the matrix function $g(\tau L)$ exactly when L is diagonalizable by the discrete cosine or sine transform (DCT/DST), cf. [Strang, 1999]. A typical case, where the DCT can be used is when L stems from the space discretization of a second-order partial differential operator with homogeneous Neumann/Dirichlet boundary conditions. In Chapter 7 we will encounter an example, where such a situation arises. ◇

5.2.2 Krylov subspace methods

If L is sparse, the multiplication with L typically requires $\mathcal{O}(m)$ operations. In such situations, iterative methods like polynomial or rational Krylov subspace methods are better suited to compute $g(\tau L)\mathbf{E}$. Here, we discuss iterative methods for computing products of the form

$$g(\tau L)\mathbf{E}, \qquad L \in \mathbb{C}^{m\times m}, \quad \mathbf{E} \in \mathbb{C}^{m\times r}. \tag{5.10}$$

Since

$$g(\tau L)\mathbf{E} = \begin{bmatrix} g(\tau L)\mathbf{E}_1 & \cdots & g(\tau L)\mathbf{E}_r \end{bmatrix},$$

where $\mathbf{E}_j$ denotes the jth column of $\mathbf{E}$, all columns of the product (5.10) can be computed independently. It hence suffices to consider products of the form

$$g(\tau L)v, \qquad v \in \mathbb{C}^m. \tag{5.11}$$

Krylov subspace methods have been studied extensively for computing approximations to (5.11) and have been shown to yield reliable results with reasonable effort. Their application is of special interest if the matrix L is sparse, since in this case a multiplication with L is cheap. The following review on Krylov subspace approximations is based on [Hochbruck et al., 2015] and [Göckler, 2014].

Polynomial Krylov subspace methods search for an approximation to the product (5.11) in the Krylov subspace

$$\mathcal{K}_\ell(L,v) = \operatorname{span}\{v, Lv, \dots, L^{\ell-1}v\} = \{p(L) \mid p \in \mathcal{P}_{\ell-1}\}, \qquad \ell \geq 1,$$

where we denote by $\mathcal{P}_\ell$ the space of polynomials with degree less than or equal to ℓ. The approximation of (5.11) proceeds in two steps: First, an orthonormal basis of $\mathcal{K}_\ell$ is computed, which can be done by the Arnoldi procedure described in Algorithm 11. Given an orthonormal basis V_ℓ, the new basis vector $v_{\ell+1}$ is derived from

$$\widetilde{v}_{\ell+1} = Lv_\ell - \sum_{j=1}^{\ell} h_{j,\ell} v_\ell, \qquad v_{\ell+1} = \frac{1}{h_{\ell+1,\ell}}\widetilde{v}_{\ell+1}, \qquad h_{\ell+1,\ell} = \|\widetilde{v}_{\ell+1}\|.$$

By construction, $v_j^H v_{\ell+1} = 0$ for $j = 1,\dots,\ell$, hence the coefficients satisfy $h_{j,\ell} = v_j^H A v_\ell$. Collecting the coefficients in the matrix $H_\ell = (h_{i,j})_{i,j=1}^{\ell} \in \mathbb{C}^{\ell\times\ell}$ we thus have the relation

$$LV_\ell = V_\ell H_\ell + h_{\ell+1,\ell} v_{\ell+1} e_\ell^T, \qquad V_\ell^H V_\ell = I_\ell, \quad V_\ell \in \mathbb{C}^{m\times\ell},$$

where e_ℓ denotes the ℓth standard basis vector of $\mathbb{R}^m$. H_ℓ is an unreduced upper Hessenberg matrix, cf. Definition A.19. Since V_ℓ is orthonormal, it is

$$\begin{aligned} V_\ell^H L V_\ell &= V_\ell^H (V_\ell H_\ell + h_{\ell+1,\ell} v_{\ell+1} e_\ell^T) \\ &= H_\ell + h_{\ell+1,\ell} \underbrace{V_\ell^H v_{\ell+1}}_{=0} e_\ell^T \\ &= H_\ell. \end{aligned}$$

However, $V_\ell V_\ell^H \neq I_m$ if $\ell < m$, and hence, in general,

$$L \neq V_\ell V_\ell^H L V_\ell V_\ell^H = V_\ell H_\ell V_\ell^H.$$

Yet the matrix $V_\ell H_\ell V_\ell^H$ approximates the argument L of the matrix function in (5.11). More precisely, H_ℓ is the projection of L onto the subspace $\mathcal{K}_\ell$. This motivates the approximation

$$g(\tau L)v \approx g(\tau V_\ell H_\ell V_\ell^H)v = V_\ell g(\tau H_\ell) V_\ell^H v, \tag{5.12}$$

where the last equality holds by Theorem A.21. Sometimes, already for $\ell \ll m$, the approximation (5.12) is sufficiently accurate. This is advantageous, since then the function g only has to be evaluated at some small matrix H_ℓ instead of the large matrix $L \in \mathbb{C}^{m\times m}$. This can be done by algorithms derived for dense matrices, e.g., by diagonalization, cf. [Higham, 2008].

Algorithm 11: Arnoldi process

```
   Input  : $L \in \mathbb{C}^{m\times m}$, $v \in \mathbb{C}^m$
1  $v_1 = v/\|v\|$
2  for $\ell = 1, 2, \dots$ do
3      for $j = 1, \dots, \ell$ do
4          $h_{j,\ell} = v_j^H L v_\ell$
5      end
6      $\widetilde{v}_{\ell+1} = L v_\ell - \sum_{j=1}^{\ell} h_{j,\ell} v_j$
7      $h_{\ell+1,\ell} = \|\widetilde{v}_{\ell+1}\|$
8      $v_{\ell+1} = \widetilde{v}_{\ell+1}/h_{\ell+1,\ell}$
9  end
```

We adapt the implementation of the polynomial Krylov subspace as provided in [Hochbruck et al., 2015, Algorihm 1], which includes several further ideas. E.g., the index ℓ is prohibited to exceed the number `maxiter` $\in \mathbb{N}$. This ensures that the Hessenberg matrix H_ℓ remains small compared to L for appropriately chosen `maxiter`.

In [Hochbruck and Lubich, 1997] it was shown that the error of the Krylov approximations to the matrix exponential always decays superlinearly, and that this decay starts at $\ell \approx \tau\|L\|_2$ iteration steps for discretizations of hyperbolic problems. Hence, if the norm of L is large and `maxiter` is chosen too small, this regime is not reached. Then, one either has to reduce the step size τ (i.e., by adding a substepping algorithm, see, e.g., [Al-Mohy and Higham, 2011]), or some kind of restarting procedure has to be used, cf. [Eiermann et al., 2011].

Another remedy to this shortcoming is to use *rational Krylov methods* instead of the polynomial variant. They have been developed in [Ruhe, 1984] and are based on the rational Krylov subspace

$$\mathcal{Q}_\ell(L,v) \coloneqq q_{\ell-1}(L)^{-1}\mathcal{K}_\ell(L,v) = \{s(L)v \mid s \in \frac{\mathcal{P}_{\ell-1}}{q_{\ell-1}}\}, \qquad \ell \geq 1,$$

where the polynomial $q_{\ell-1} \in \mathcal{P}_{\ell-1}$ is assumed to have no roots in the spectrum of L. This ensures that the inverse of $q_{\ell-1}(L)$ exists. With Theorem A.21, one obtains the alternative representation

$$\mathcal{Q}_\ell(L,v) = \mathcal{K}_\ell(L, q_{\ell-1}(L)^{-1}v).$$

The properties of the rational Krylov subspace approximation strongly depend on the polynomial $q_{\ell-1}$. We restrict ourselves to the particular choice

$$q_{\ell-1}(z) = (\gamma - z)^{\ell-1}, \qquad \gamma > 0,$$

which yields the so-called *shift-and-invert Krylov subspace* $\mathcal{Q}_\ell(L,v)$ with

$$\mathcal{Q}_\ell(L,v) = \mathcal{K}_\ell\big((\gamma I - \tau L)^{-1}, v\big) = \operatorname{span}\left\{v, (\gamma I - \tau L)^{-1}v, \dots, (\gamma I - \tau L)^{-(\ell-1)}v\right\}.$$

The Arnoldi algorithm now computes an orthonormal basis V_ℓ and an upper Hessenberg matrix H_ℓ such that

$$(\gamma I - \tau L)^{-1}V_\ell = V_\ell H_\ell + h_{\ell+1,\ell}v_{\ell+1}e_\ell^T, \qquad V_\ell^H V_\ell = I_\ell, \tag{5.13}$$

or equivalently

$$(\tau L - \gamma I)V_\ell H_\ell = -V_\ell + (\gamma I - \tau L)h_{\ell+1,\ell}v_{\ell+1}e_\ell^T. \tag{5.14}$$

Multiplication of (5.14) with V_ℓ^H from the left yields

$$\begin{aligned}
& V_\ell^H(\tau L - \gamma I)V_\ell H_\ell = -V_\ell^H V_\ell + V_\ell^H(\gamma I - \tau L)h_{\ell+1,\ell}v_{\ell+1}e_\ell^T \\
\iff \quad & V_\ell^H(\tau L)V_\ell H_\ell = -I + \gamma H_\ell + \gamma h_{\ell+1,\ell}\underbrace{V_\ell^H v_{\ell+1}}_{=0}e_\ell^T - h_{\ell+1,\ell}V_\ell^H(\tau L)v_{\ell+1}e_\ell^T \\
\iff \quad & \widehat{H}_\ell \coloneqq V_\ell^H(\tau L)V_\ell = \gamma I - H_\ell^{-1} - h_{\ell+1,\ell}V_\ell^H(\tau L)v_{\ell+1}e_\ell^T H_\ell^{-1}
\end{aligned} \tag{5.15}$$

Similarly as for the polynomial case, we end up with the rational approximation

$$g(\tau L)v \approx V_\ell g(\widehat{H}_\ell)V_\ell^H v$$

to (5.11).

In [Hochbruck et al., 2015, Algorithm 2], we find an implementation of the shift-and-invert Krylov method, which we adapt for our numerical experiments. It was pointed out in [Hochbruck et al., 2015, p. 250], that the last term in (5.15) can be neglected. Either way, in each step of the method linear systems of equations have to be solved, cf. (5.13). In practice, these are not solved exactly but again with some iterative method, e.g., a preconditioned Krylov subspace method. For the special case of L being a Toeplitz matrix, cf. Definition A.17, we do this as presented in [Lee et al., 2010]. There the authors exploit the special structure admitted by Toeplitz matrices in order to derive a fast and reliable variant of the shift-and-invert Krylov subspace method tailored to this case. The combination of their approach and [Hochbruck et al., 2015, Algorithm 2] is used in our numerical experiments in order to evaluate (5.11) for this particular case.

5.3 Implementation of the low-rank Gautschi method

We now develop an efficient implementation of the low-rank Gautschi method introduced in Section 4.4.2, and show how to compute the low-rank approximations $\mathbf{A}_1$ and $\mathbf{B}_1$ in (4.61). Recall that Ω is symmetric and positive definite, hence its eigenvalues are real. In the following, we restrict ourselves to the case, where Ω is Fourier diagonalizable, i.e.,

$$\Omega = \mathcal{F}_m^{-1} \Lambda \mathcal{F}_m, \qquad \Lambda \in \mathbb{R}^{m\times m} \text{ diagonal}, \quad \mathcal{F}_m \text{ discrete Fourier transform matrix.} \tag{5.16}$$

Remark 5.3. 1. Throughout, for any $A \in \mathbb{C}^{m\times n}$ we call the matrix $\mathcal{F}_m A$ living in Fourier space, while A lives in physical space. The discrete Fourier-transform operator $\mathcal{F}_m$ hence transforms from physical space into the Fourier space, respectively, and its inverse $\mathcal{F}_m^{-1}$ into the opposite direction.

2. For simplicity, we will treat the discrete Fourier-transform as a matrix in all calculations in this section. Note that $\mathcal{F}_m^{-H} = \frac{1}{m}\mathcal{F}_m$ and

$$\mathcal{F}_m^H \mathcal{F}_m = m\mathcal{F}_m^{-1}\mathcal{F}_m = mI, \tag{5.17}$$

so that $\mathcal{F}_m$ is orthogonal, but not unitary. $\diamond$

The main difficulty in the implementation of the low-rank Gautschi method is imposed by the filter functions. In particular, the right-hand side f is not evaluated at some low-rank approximation $\mathbf{A}_k \approx A(t_k)$, but the filtered counterpart $\mathbf{\Phi}\mathbf{A}_k$ according to (4.59).

Due to (5.16), we compute the matrix functions in (4.57), (4.58), and (4.59) exactly by (5.9):

$$f(\tau\Omega) = \mathcal{F}_m^{-1} f(\tau\Lambda)\mathcal{F}_m, \qquad f(\xi) \in \{\cos(\xi), \operatorname{sinc}(\xi), -\xi^2 \operatorname{sinc}(\xi), \psi(\xi), \psi_0(\xi), \psi_1(\xi), \phi(\xi)\}.$$

We abbreviate

$$\boldsymbol{\mathcal{A}}_k = \mathcal{F}_m \mathbf{A}_k, \quad \boldsymbol{\mathcal{B}}_k = \mathcal{F}_m \mathbf{B}_k, \qquad k \in \mathbb{N}_0.$$

Then, the update steps of the full-rank Gautschi scheme read

$$A_{k+1} = \mathcal{F}_m^{-1}\big(\cos(\tau\Lambda)\boldsymbol{\mathcal{A}}_k + \tau \operatorname{sinc}(\tau\Lambda)\boldsymbol{\mathcal{B}}_k + \frac{\tau^2}{2}\psi(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k)\big), \tag{5.18a}$$

$$\begin{aligned} B_{k+1} = \mathcal{F}_m^{-1}\big(&-\tau\Lambda^2 \operatorname{sinc}(\tau\Lambda)\boldsymbol{\mathcal{A}}_k + \cos(\tau\Lambda)\boldsymbol{\mathcal{B}}_k \\ &+ \frac{\tau}{2}(\psi_0(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k) + \psi_1(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_{k+1}))\big). \end{aligned} \tag{5.18b}$$

Hence, for computing A_{k+1} and B_{k+1}, 6 and 5 (inverse) FFTs are required, respectively. Note, that the matrix $\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k)$ can be reused from the A-step. We now focus on the numerous (inverse) Fourier transformations in (5.18). We want to compute low-rank approximations $\mathbf{A}_k$ and $\mathbf{B}_k$ to the exact solution of (4.56) with as few (inverse) Fourier transformations as possible. The crucial idea is to compute low-rank approximations in Fourier space rather than in physical space. This is possible, since the Fourier transformation preserves the rank:

$$\mathbf{A} \in \mathcal{M}_r \iff \mathcal{F}_m\mathbf{A} \in \mathcal{M}_r \iff \mathcal{F}_m^{-1}\mathbf{A} \in \mathcal{M}_r.$$

We conclude, that $\mathcal{F}_m\mathbf{A}$ admits a low-rank decomposition into factors $\boldsymbol{\mathcal{U}} \in \mathcal{V}_{m,r}$, $\boldsymbol{\mathcal{S}} \in \mathbb{C}^{r\times r}$, and $\boldsymbol{\mathcal{V}} \in \mathcal{V}_{n,r}$. By the associativity of the matrix product, we have

$$\mathcal{F}_m\mathbf{A} = \mathcal{F}_m(\mathbf{USV}^H) = (\mathcal{F}_m\mathbf{U})\mathbf{SV}^H.$$

This already resembles the desired factorization, but the factor $\mathcal{F}_m\mathbf{U}$ is not contained in the Stiefel manifold $\mathcal{V}_{m,r}$, since by (5.17) we have

$$(\mathcal{F}_m\mathbf{U})^H(\mathcal{F}_m\mathbf{U}) = \mathbf{U}^H\mathcal{F}_m^H\mathcal{F}_m\mathbf{U} = m\mathbf{U}^H\mathbf{U} = mI.$$

However, by rescaling the factors we get

$$\mathcal{F}_m\mathbf{A} = \big(\frac{1}{\sqrt{m}}\mathcal{F}_m\mathbf{U}\big)(\sqrt{m}\mathbf{S})\mathbf{V}^H = \boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{S}}\boldsymbol{\mathcal{V}}^H, \qquad \text{where } \boldsymbol{\mathcal{U}} \in \mathcal{V}_{m,r}, \quad \boldsymbol{\mathcal{S}} \in \mathbb{C}^{r\times r}, \quad \boldsymbol{\mathcal{V}} \in \mathcal{V}_{n,r}. \tag{5.19}$$

One can compute a low-rank factorization of the Fourier transformation of some low-rank matrix by the projector-splitting integrator in Fourier space:

Lemma 5.4. *Let $A(t) \in \mathbb{C}^{m\times n}$ be a time-dependent matrix of arbitrary rank, $\mathbf{A}_k \approx A(t_k)$ a rank-r approximation with the factorization $\mathbf{A}_k = \mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^H \in \mathcal{M}_r$ and $A_{k+1} \approx A(t_{k+1})$ an arbitrary approximation computed with a standard scheme. Denote by $\mathbf{A}_{k+1} \in \mathcal{M}_r$ the low-rank approximation to $A(t_{k+1})$ computed by the projector-splitting integrator with increment $\Delta A = A_{k+1} - \mathbf{A}_k$.*

Then $\boldsymbol{\mathcal{A}}_{k+1} = \boldsymbol{\mathcal{U}}_{k+1}\boldsymbol{\mathcal{S}}_{k+1}\boldsymbol{\mathcal{V}}_{k+1}^H$, where the factors $\boldsymbol{\mathcal{U}}_{k+1}, \boldsymbol{\mathcal{S}}_{k+1}$, and $\boldsymbol{\mathcal{V}}_{k+1}$ result from one step of the projector-splitting integrator, cf. Algorithm 2, with increment $\mathcal{F}_m\Delta A$.

Proof. Without loss of generality we consider $k = 0$. We recall the update steps of the projector-splitting integrator for computing $\mathbf{A}_1 \approx A(\tau)$:

$$\mathbf{U}_1\widehat{\mathbf{S}}_1 = \mathbf{U}_0\mathbf{S}_0 + (A_1 - \mathbf{A}_0)\mathbf{V}_0 = A_1\mathbf{V}_0, \tag{5.20a}$$

$$\widetilde{\mathbf{S}}_0 = \widehat{\mathbf{S}}_1 - \mathbf{U}_1^H(A_1 - \mathbf{A}_0)\mathbf{V}_0 = \mathbf{U}_1^H A_1\mathbf{V}_0 - \mathbf{U}_1^H A_1\mathbf{V}_0 + \mathbf{U}_1^H\mathbf{U}_0\mathbf{S}_0\mathbf{V}_0^H\mathbf{V}_0 = \mathbf{U}_1^H\mathbf{U}_0\mathbf{S}_0, \tag{5.20b}$$

$$\mathbf{V}_1\mathbf{S}_1^H = \mathbf{V}_0\widetilde{\mathbf{S}}_0^H + (A_1 - \mathbf{A}_0)^H\mathbf{U}_1 = \mathbf{V}_0\mathbf{S}_0^H\mathbf{U}_0^H\mathbf{U}_1 + A_1^H\mathbf{U}_1 - \mathbf{V}_0\mathbf{S}_0^H\mathbf{U}_0^H\mathbf{U}_1 = A_1^H\mathbf{U}_1. \tag{5.20c}$$

In Fourier space, we first need a factorization of the Fourier transformation $\boldsymbol{\mathcal{A}}_0$ of $\mathbf{A}_0$. By (5.19),

$$\boldsymbol{\mathcal{A}}_0 = \boldsymbol{\mathcal{U}}_0\boldsymbol{\mathcal{S}}_0\boldsymbol{\mathcal{V}}_0^H = \big(\frac{1}{\sqrt{m}}\mathcal{F}_m\mathbf{U}_0\big)(\sqrt{m}\mathbf{S}_0)\mathbf{V}_0^H.$$

The first substep of the projector-splitting integrator with increment $\mathcal{F}_m A_1 - \boldsymbol{\mathcal{A}}_0$ reads

$$\boldsymbol{\mathcal{U}}_1\widehat{\boldsymbol{\mathcal{S}}}_1 = \boldsymbol{\mathcal{U}}_0\boldsymbol{\mathcal{S}}_0 + (\mathcal{F}_m A_1 - \boldsymbol{\mathcal{A}}_0)\mathbf{V}_0 = \mathcal{F}_m A_1\mathbf{V}_0.$$

Replacing $A_1\mathbf{V}_0$ by (5.20a) yields

$$\boldsymbol{\mathcal{U}}_1\widehat{\boldsymbol{\mathcal{S}}}_1 = \mathcal{F}_m(\mathbf{U}_1\widehat{\mathbf{S}}_1) = \big(\frac{1}{\sqrt{m}}\mathcal{F}_m\mathbf{U}_1\big)(\sqrt{m}\widehat{\mathbf{S}}_1).$$

Since the QR-decomposition is unique up to a unitary diagonal matrix, there exists $D_1 \in \mathbb{C}^{r\times r}$ such that

$$\boldsymbol{\mathcal{U}}_1 = \frac{1}{\sqrt{m}}\mathcal{F}_m\mathbf{U}_1 D_1, \qquad \widehat{\boldsymbol{\mathcal{S}}}_1 = \sqrt{m}D_1^H\widehat{\mathbf{S}}_1,$$

cf. Section A.2. For the second substep in Fourier space, we have

$$\begin{aligned}\widetilde{\boldsymbol{\mathcal{S}}}_0 &= \widehat{\boldsymbol{\mathcal{S}}}_1 - \boldsymbol{\mathcal{U}}_1^H(\mathcal{F}_m A_1 - \boldsymbol{\mathcal{A}}_0)\mathbf{V}_0 \\ &= \boldsymbol{\mathcal{U}}_1^H\mathcal{F}_m A_1\mathbf{V}_0 - \boldsymbol{\mathcal{U}}_1^H\mathcal{F}_m A_1\mathbf{V}_0 + \boldsymbol{\mathcal{U}}_1^H\mathcal{F}_m\mathbf{A}_0\mathbf{V}_0 \\ &= \frac{1}{\sqrt{m}}D_1^H\mathbf{U}_1^H\mathcal{F}_m^H\mathcal{F}_m\mathbf{U}_0\mathbf{S}_0 \\ &= \sqrt{m}D_1^H\mathbf{U}_1^H\mathbf{U}_0\mathbf{S}_0 \\ &= \sqrt{m}D_1^H\widetilde{\mathbf{S}}_0,\end{aligned}$$

where we replaced $\mathbf{U}_1^H \mathbf{U}_0 \mathbf{S}_0$ by (5.20b) in the last step. Finally, the third substep yields

$$\begin{aligned} \boldsymbol{\mathcal{V}}_1 \boldsymbol{\mathcal{S}}_1^H &= \mathbf{V}_0 \widetilde{\boldsymbol{\mathcal{S}}}_0^H + (\mathcal{F}_m A_1 - \boldsymbol{\mathcal{A}}_0)^H \boldsymbol{\mathcal{U}}_1 \\ &= \sqrt{m} \mathbf{V}_0 \mathbf{S}_0^H \mathbf{U}_0^H \mathbf{U}_1 D_1 + \frac{1}{\sqrt{m}} A_1^H \mathcal{F}_m^H \mathcal{F}_m \mathbf{U}_1 D_1 - \frac{1}{\sqrt{m}} \mathbf{A}_0^H \mathcal{F}_m^H \mathcal{F}_m \mathbf{U}_1 D_1 \\ &= \sqrt{m} \mathbf{A}_1^H \mathbf{U}_1 D_1 \\ &= \mathbf{V}_1 (\sqrt{m} D_1^H \mathbf{S}_1)^H \end{aligned}$$

by (5.20c). Again, there exists a unitary diagonal matrix $D_2 \in \mathbb{C}^{r \times r}$ with

$$\boldsymbol{\mathcal{V}}_1 = \mathbf{V}_1 D_2 \qquad \boldsymbol{\mathcal{S}}_1 = \sqrt{m} D_1^H \mathbf{S}_1 D_2.$$

Altogether, we thus have

$$\begin{aligned} \boldsymbol{\mathcal{U}}_1 \boldsymbol{\mathcal{S}}_1 \boldsymbol{\mathcal{V}}_1^H &= \big(\frac{1}{\sqrt{m}} \mathcal{F}_m \mathbf{U}_1 D_1\big)(\sqrt{m} D_1^H \mathbf{S}_1 D_2)(\mathbf{V}_1 D_2)^H \\ &= \mathcal{F}_m \mathbf{U}_1 D_1 D_1^H \mathbf{S}_1 D_2 D_2^H \mathbf{V}_1^H \\ &= \mathcal{F}_m \mathbf{A}_1 = \boldsymbol{\mathcal{A}}_1, \end{aligned}$$

which proves the claim. Note that we can employ the modified update sequence (4.14) also in Fourier space. □

The impact of Lemma 5.4 is considerable: Denote by A_{k+1}, B_{k+1} the approximations to $A(t_{k+1})$ and $A'(t_{k+1})$ computed with a single step of the Gautschi scheme (4.58) started from the low-rank approximations $\mathbf{A}_k, \mathbf{B}_k$. If we want to compute a low-rank approximation $\mathbf{A}_{k+1}$ to A_{k+1}, we can instead compute a low-rank approximation $\boldsymbol{\mathcal{A}}_{k+1} \approx \mathcal{F}_m A_{k+1}$, and similarly for B_{k+1}:

$$\boldsymbol{\mathcal{A}}_{k+1} \approx \mathcal{F}_m A_{k+1} = \cos(\tau\Lambda) \boldsymbol{\mathcal{A}}_k + \tau \operatorname{sinc}(\tau\Lambda) \boldsymbol{\mathcal{B}}_k + \frac{\tau^2}{2} \psi(\tau\Lambda) \mathcal{F}_m f\big(\mathcal{F}_m^{-1} \phi(\tau\Lambda) \boldsymbol{\mathcal{A}}_k\big), \tag{5.21a}$$

$$\begin{aligned} \boldsymbol{\mathcal{B}}_{k+1} \approx \mathcal{F}_m B_{k+1} = &-\tau\Lambda^2 \operatorname{sinc}(\tau\Lambda) \boldsymbol{\mathcal{A}}_k + \cos(\tau\Lambda) \boldsymbol{\mathcal{B}}_k \\ &+ \frac{\tau}{2} (\psi_0(\tau\Lambda) \mathcal{F}_m f(\mathcal{F}_m^{-1} \phi(\tau\Lambda) \boldsymbol{\mathcal{A}}_k) + \psi_1(\tau\Lambda) \mathcal{F}_m f(\mathcal{F}_m^{-1} \phi(\tau\Lambda) \boldsymbol{\mathcal{A}}_{k+1})). \end{aligned} \tag{5.21b}$$

The representations of $\boldsymbol{\mathcal{A}}_{k+1}$ and $\boldsymbol{\mathcal{B}}_{k+1}$ in terms of the matrices $\boldsymbol{\mathcal{A}}_k$, $\boldsymbol{\mathcal{B}}_k$, and $\boldsymbol{\mathcal{A}}_{k+1}$ reveal that a low-rank approximation in Fourier space is sufficient to compute the next step of the low-rank Gautschi method. If

$$f(\mathcal{F}_m^{-1} A) = \mathcal{F}_m^{-1} f(A)$$

is true for each $A \in \mathbb{C}^{m \times n}$, it is possible to carry out the whole computation in Fourier space. Then one could first transform the initial values $\mathbf{A}_0 \in \mathcal{M}_{r_\mathbf{A}}$ and $\mathbf{B}_0 \in \mathcal{M}_{r_\mathbf{B}}$ into Fourier space, iterate k times and transform the approximations back to physical space with the inverse Fourier transform.

Given decompositions $\boldsymbol{\mathcal{A}}_k = \boldsymbol{\mathcal{U}}_k \boldsymbol{\mathcal{S}}_k \boldsymbol{\mathcal{V}}_k^H \in \mathcal{M}_{r_\mathbf{A}}$ and $\boldsymbol{\mathcal{B}}_k = \boldsymbol{\mathcal{T}}_k \boldsymbol{\mathcal{R}}_k \boldsymbol{\mathcal{W}}_k^H \in \mathcal{M}_{r_\mathbf{B}}$, recall that we have to compute the products

$$\mathcal{F}_m A_{k+1} \boldsymbol{\mathcal{V}}_k, \quad (\mathcal{F}_m A_{k+1})^H \boldsymbol{\mathcal{U}}_{k+1}, \quad \mathcal{F}_m B_{k+1} \boldsymbol{\mathcal{W}}_k, \quad \text{and} \quad (\mathcal{F}_m B_{k+1})^H \boldsymbol{\mathcal{T}}_{k+1}, \tag{5.22}$$

where $\mathcal{F}_m A_{k+1}$ is given in (5.21a) and $\mathcal{F}_m B_{k+1}$ in (5.21b), respectively, in order to compute the low-rank approximations $\boldsymbol{\mathcal{A}}_{k+1}$ and $\boldsymbol{\mathcal{B}}_{k+1}$. In the following, we explain in detail how to compute these products. For this, assume we are already given the matrices

$$\mathbf{L}_\mathbf{A} = \boldsymbol{\mathcal{V}}_k \boldsymbol{\mathcal{S}}_k^H, \qquad \mathbf{L}_\mathbf{B} = \boldsymbol{\mathcal{W}}_k \boldsymbol{\mathcal{R}}_k^H, \qquad \widetilde{\boldsymbol{\mathcal{U}}} = \mathcal{F}_m^{-1} \phi(\tau\Lambda) \boldsymbol{\mathcal{U}}_k \tag{5.23}$$

from the previous step of the iteration. Moreover, we also assume that the diagonal matrices

$$f(\tau\Lambda), \qquad ff(\xi) \in \{\cos(\xi), \operatorname{sinc}(\xi), -\xi^2 \operatorname{sinc}(\xi), \psi(\xi), \psi_0(\xi), \psi_1(\xi), \phi(\xi)\}$$

have been computed and stored beforehand. Lastly, for simplicity we only consider the case $r_{\mathbf{A}} = r_{\mathbf{B}} = r$. The extension to $r_{\mathbf{A}} \neq r_{\mathbf{B}}$ is straightforward.

We start the current $\boldsymbol{\mathcal{A}}$-step by firstly computing and storing the matrices

$$\mathbf{Y}_U = \boldsymbol{\mathcal{U}}_k \boldsymbol{\mathcal{S}}_k \in \mathbb{C}^{m\times r}, \qquad \mathbf{Y}_T = \boldsymbol{\mathcal{T}}_k \boldsymbol{\mathcal{R}}_k \in \mathbb{C}^{n\times r}, \qquad \mathbf{X} = \boldsymbol{\mathcal{W}}_k^H \boldsymbol{\mathcal{V}}_k \in \mathbb{C}^{r\times r}.$$

This allows us to compute the matrix

$$\mathbf{K} = \cos(\tau\Lambda)\mathbf{Y}_U + \tau \operatorname{sinc}(\tau\Lambda)\mathbf{Y}_T\mathbf{X} \tag{5.24}$$

by successive matrix multiplication from right to left, and matrix addition. It remains to compute

$$\frac{\tau^2}{2}\psi(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k)\boldsymbol{\mathcal{V}}_k$$

for finalizing the first product in (5.22). As we have seen in Section 5.1, for the evaluation of the right-hand side f, a decomposition of the argument of f into factors $\widetilde{\mathbf{U}} \in \mathbb{C}^{m\times r}$ and $\widetilde{\mathbf{L}} \in \mathbb{C}^{n\times r}$ is mandatory. However, apart from their dimension, no further properties are imposed on these factors. Per construction, $\boldsymbol{\mathcal{A}}_k$ admits the decomposition $\boldsymbol{\mathcal{A}}_k = \boldsymbol{\mathcal{U}}_k \boldsymbol{\mathcal{S}}_k \boldsymbol{\mathcal{V}}_k^H$, hence the product $f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k)\boldsymbol{\mathcal{V}}_k$ can be computed as presented in Section 5.1, with the factors

$$\widetilde{\mathbf{U}} = \mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{U}}_k, \qquad \widetilde{\mathbf{L}} = \boldsymbol{\mathcal{V}}_k \boldsymbol{\mathcal{S}}_k^H.$$

Comparison to (5.23) reveals, that $\widetilde{\mathbf{U}} = \widetilde{\boldsymbol{\mathcal{U}}}$ and $\widetilde{\mathbf{L}} = \mathbf{L}_{\mathbf{A}}$, i.e., both quantities do not have to be computed but are already available from the previous iteration step. Overall, this enables us to compute the product

$$\mathbf{Y} = f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k)\boldsymbol{\mathcal{V}}_k = f(\widetilde{\boldsymbol{\mathcal{U}}}\mathbf{L}_{\mathbf{A}}^H)\boldsymbol{\mathcal{V}}_k$$

efficiently. The matrix $\mathbf{K}$ in (5.24) is then updated to

$$\mathbf{K} \leftarrow \mathbf{K} + \frac{\tau^2}{2}\psi(\tau\Lambda)\mathcal{F}_m\mathbf{Y},$$

where we compute the arising products again from right to left. Overall, this yields

$$\mathbf{K} = \mathcal{F}_m A_{k+1}\boldsymbol{\mathcal{V}}_k = \boldsymbol{\mathcal{U}}_{k+1}\boldsymbol{\mathcal{S}} \qquad \text{by } QR\text{-decomposition.} \tag{5.25}$$

Remark 5.5. Observe that reusing $\widetilde{\boldsymbol{\mathcal{U}}}$ in (5.23) saved one inverse Fourier transform. Thus, for computing $\mathbf{K}$ in (5.25) only a single Fourier transformation is required. ◇

The second product in (5.22) is now performed similarly. We first have

$$\mathbf{L}_{\mathbf{A}}^1 = \mathbf{L}_{\mathbf{A}}\boldsymbol{\mathcal{U}}_k^H\cos(\tau\Lambda)\boldsymbol{\mathcal{U}}_{k+1} + \tau\mathbf{L}_{\mathbf{B}}\boldsymbol{\mathcal{T}}_k^H\operatorname{sinc}(\tau\Lambda)\boldsymbol{\mathcal{U}}_{k+1}.$$

Secondly, it is

$$\begin{aligned}\mathbf{Z} &= \big(\psi(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k)\big)^H\boldsymbol{\mathcal{U}}_{k+1}\\ &= f(\widetilde{\boldsymbol{\mathcal{U}}}\mathbf{L}_{\mathbf{A}}^H)^H(\mathcal{F}_m)^H\psi(\tau\Lambda)^H\boldsymbol{\mathcal{U}}_{k+1}\\ &= m f(\widetilde{\boldsymbol{\mathcal{U}}}\mathbf{L}_{\mathbf{A}}^H)^H(\mathcal{F}_m^{-1}\psi(\tau\Lambda)\boldsymbol{\mathcal{U}}_{k+1}).\end{aligned}$$

We thus can again make use of the efficient routines of Section 5.1 for the computation of $\mathbf{Z}$. $\mathbf{L}^1_{\mathbf{A}}$ is then updated as

$$\mathbf{L}^1_{\mathbf{A}} \leftarrow \mathbf{L}^1_{\mathbf{A}} + \frac{\tau^2}{2}\mathbf{Z}.$$

Computing the QR-decomposition of $\mathbf{L}^1_{\mathbf{A}}$ yields the factors $\boldsymbol{\mathcal{V}}_{k+1}$ and $\boldsymbol{\mathcal{S}}^H_{k+1}$. This finalizes the $\boldsymbol{\mathcal{A}}$-step of the low-rank Gautschi method.

We now turn to the $\boldsymbol{\mathcal{B}}$-step of the scheme. Both $\boldsymbol{\mathcal{A}}_k$ and $\boldsymbol{\mathcal{A}}_{k+1}$ are needed to compute the step, but since we computed the $\boldsymbol{\mathcal{A}}$-step completely, we have both matrices available in means of their factors. We first compute

$$\widehat{\mathbf{K}} = -\tau\Lambda^2 \operatorname{sinc}(\tau\Lambda)\mathbf{Y}_U\mathbf{X}^H + \cos(\tau\Lambda)\mathbf{Y}_T,$$

again from right to left. Next, we set

$$\widetilde{\boldsymbol{\mathcal{U}}}_1 = \mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{U}}_{k+1}.$$

Then the product

$$\begin{aligned}\widehat{\mathbf{Y}} &= (\psi_0(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k) + \psi_1(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_{k+1}))\boldsymbol{\mathcal{W}}_k \\ &= \psi_0(\tau\Lambda)\mathcal{F}_m f(\widetilde{\boldsymbol{\mathcal{U}}}\mathbf{L}^H_{\mathbf{A}})\boldsymbol{\mathcal{W}}_k + \psi_1(\tau\Lambda)\mathcal{F}_m f(\widetilde{\boldsymbol{\mathcal{U}}}_1(\mathbf{L}^1_{\mathbf{A}})^H)\boldsymbol{\mathcal{W}}_k\end{aligned}$$

can be performed analogously as before for the $\boldsymbol{\mathcal{A}}$-step based on the ansatz in Section 5.1. Updating $\widehat{\mathbf{K}} \leftarrow \widehat{\mathbf{K}} + \widehat{\mathbf{Y}}$ and computing the QR-decomposition $\widehat{\mathbf{K}} = \boldsymbol{\mathcal{T}}_{k+1}\boldsymbol{\mathcal{R}}$ yields the factor $\boldsymbol{\mathcal{T}}_{k+1}$ of $\boldsymbol{\mathcal{B}}_{k+1}$. Lastly, with

$$\mathbf{L}^1_{\mathbf{B}} = -\tau^2\boldsymbol{\mathcal{V}}_k\mathbf{Y}_U^H\Lambda^2\operatorname{sinc}(\tau\Lambda)\boldsymbol{\mathcal{T}}_{k+1} + \boldsymbol{\mathcal{W}}_k\mathbf{Y}_T^H\cos(\tau\Lambda)\boldsymbol{\mathcal{T}}_{k+1}$$

and

$$\begin{aligned}\widehat{\mathbf{Z}} &= (\psi_0(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_k) + \psi_1(\tau\Lambda)\mathcal{F}_m f(\mathcal{F}_m^{-1}\phi(\tau\Lambda)\boldsymbol{\mathcal{A}}_{k+1}))^H\boldsymbol{\mathcal{T}}_k \\ &= m\big(f(\widetilde{\boldsymbol{\mathcal{U}}}\mathbf{L}^H_{\mathbf{A}})^H(\mathcal{F}_m^{-1}\psi_0(\tau\Lambda)\boldsymbol{\mathcal{T}}_{k+1}) + f(\widetilde{\boldsymbol{\mathcal{U}}}_1(\mathbf{L}^1_{\mathbf{A}})^H)^H(\mathcal{F}_m^{-1}\psi_1(\tau\Lambda)\boldsymbol{\mathcal{T}}_{k+1})\big),\end{aligned}$$

we get

$$\mathbf{L}^1_{\mathbf{B}} \leftarrow \mathbf{L}^1_{\mathbf{B}} + \widehat{\mathbf{Z}}, \qquad \mathbf{L}^1_{\mathbf{B}} = \boldsymbol{\mathcal{W}}_{k+1}\boldsymbol{\mathcal{R}}^H_{k+1} \text{ by } QR\text{-decomposition.}$$

This finalizes the $\boldsymbol{\mathcal{B}}$-step of the low-rank Gautschi method. The quantities $\widetilde{\boldsymbol{\mathcal{U}}}_1$, $\mathbf{L}^1_{\mathbf{A}}$, and $\mathbf{L}^1_{\mathbf{B}}$ are stored and reused in the subsequent full step of the iteration. Overall, we require three Fourier transformations (one for the $\boldsymbol{\mathcal{A}}$-step, two for the $\boldsymbol{\mathcal{B}}$-step) and four inverse transformations (one for the $\boldsymbol{\mathcal{A}}$-step, three for the $\boldsymbol{\mathcal{B}}$-step). This is a significant reduction of Fourier operations compared to a naive implementation based on (5.21) and the projector-splitting integrator, which requires in total twelve Fourier operations.

5.4 Splitting for second-order matrix differential equations

We return to the stiff second-order matrix differential equation (4.49),

$$A''(t) = -\Omega_1^2 A(t) - A(t)\Omega_2^2 + f\big(A(t)\big), \qquad A(0) = A_0, \quad A'(0) = B_0,$$

and consider again the splitting (4.55)

$$\begin{bmatrix} A(t) \\ B(t) \end{bmatrix}' = \begin{bmatrix} B(t) \\ -\Omega_1^2 A(t) - A(t)\Omega_2^2 + f(A(t)) \end{bmatrix} = \begin{bmatrix} \widetilde{\omega}_1^2 B(t) \\ -\Omega_1^2 A(t) - A(t)\Omega_2^2 \end{bmatrix} + \begin{bmatrix} \widetilde{\omega}_2^2 B(t) \\ f\big(A(t)\big) \end{bmatrix}, \qquad \widetilde{\omega}_1^2 + \widetilde{\omega}_2^2 = 1,$$

instead of (4.50). Here the whole linear part is separated from the nonlinear part like it is done in the construction of the variant `prsistiff` of the projector-splitting integrator, see Section 3.3. For simplicity, let $\widetilde{\omega}_1 = 1$ and $\widetilde{\omega}_2 = 0$ in the following. The exact solution to the first (linear) subproblem

$$A''(t) = -\Omega_1^2 A(t) - A(t)\Omega_2^2, \quad t \in [0,T], \qquad A(0) = \mathbf{A}_0, \quad A'(0) = \mathbf{B}_0,$$

can be given explicitly in terms of the matrix exponential. By vectorization, the above differential equation takes the form

$$\operatorname{vec} A''(t) = -\Omega^2 \operatorname{vec} A(t), \qquad \Omega^2 = \Omega_1^2 \oplus (\Omega_2^2)^T,$$

cf. Definition A.15, or equivalently in first-order formulation

$$\begin{bmatrix} \operatorname{vec} A(t) \\ \operatorname{vec} B(t) \end{bmatrix}' = \begin{bmatrix} 0 & I \\ -\Omega^2 & 0 \end{bmatrix} \begin{bmatrix} \operatorname{vec} A(t) \\ \operatorname{vec} B(t) \end{bmatrix}.$$

We immediately conclude

$$\begin{bmatrix} \operatorname{vec} A(t) \\ \operatorname{vec} B(t) \end{bmatrix} = \exp\left(t \begin{bmatrix} 0 & I \\ -\Omega^2 & 0 \end{bmatrix} \right) \begin{bmatrix} \operatorname{vec} \mathbf{A}_0 \\ \operatorname{vec} \mathbf{B}_0 \end{bmatrix} = \begin{bmatrix} \cos(t\Omega) & t\operatorname{sinc}(t\Omega) \\ -t\Omega^2 \operatorname{sinc}(t\Omega) & \cos(t\Omega) \end{bmatrix} \begin{bmatrix} \operatorname{vec} \mathbf{A}_0 \\ \operatorname{vec} \mathbf{B}_0 \end{bmatrix}.$$

Consider now the situation, that both Ω_1 and Ω_2 are Fourier-diagonalizable,

$$\Omega_1 = \mathcal{F}_m^{-1} \Lambda_1 \mathcal{F}_m, \quad \Omega_2 = \mathcal{F}_n^{-1} \Lambda_2 \mathcal{F}_n, \qquad \Lambda_1, \Lambda_2 \text{ diagonal.}$$

Then one can compute the matrix exponential exactly by diagonalization, cf. Section 5.2. We define

$$\mathcal{F} = \mathcal{F}_n^{-T} \otimes \mathcal{F}_m, \qquad \Lambda^2 = \Lambda_1^2 \oplus \Lambda_2^2 \in \mathbb{C}^{mn \times mn},$$

and note that Λ^2 is diagonal. By the rules of the Kronecker product, see Theorem A.11, we have

$$\begin{aligned} \mathcal{F}^{-1} \Lambda^2 \mathcal{F} &= (\mathcal{F}_n^T \otimes \mathcal{F}_m^{-1})(I_n \otimes \Lambda_1^2 + \Lambda_2^2 \otimes I_m)(\mathcal{F}_n^{-T} \otimes \mathcal{F}_m) \\ &= \left[\mathcal{F}_n^T \otimes \mathcal{F}_m^{-1} \Lambda_1^2 + \mathcal{F}_n^T \Lambda_2^2 \otimes \mathcal{F}_m^{-1} \right] (\mathcal{F}_n^{-T} \otimes \mathcal{F}_m) \\ &= I_n \otimes \mathcal{F}_m^{-1} \Lambda_1^2 \mathcal{F}_m + \mathcal{F}_n^T \Lambda_2^2 \mathcal{F}_n^{-T} \otimes I_m \\ &= \Omega_1^2 \oplus (\Omega_2^2)^T, \end{aligned}$$

so that we found a diagonalization of the Kronecker sum $\Omega_1^2 \oplus (\Omega_2^2)^T$. Overall, this shows that the matrix exponential above admits the alternative representation

$$\exp\left(t \begin{bmatrix} 0 & I \\ -\Omega^2 & 0 \end{bmatrix} \right) = \begin{bmatrix} \mathcal{F}^{-1} & \\ & \mathcal{F}^{-1} \end{bmatrix} \begin{bmatrix} \cos(t\Lambda) & t\operatorname{sinc}(t\Lambda) \\ -t\Lambda^2 \operatorname{sinc}(t\Lambda) & \cos(t\Lambda) \end{bmatrix} \begin{bmatrix} \mathcal{F} & \\ & \mathcal{F} \end{bmatrix}.$$

The computation of the exact solution $A(t)$ hence requires to compute and store the diagonal matrices $\cos(t\Lambda)$, $t\operatorname{sinc}(t\Lambda)$, and $-t\Lambda^2 \operatorname{sinc}(t\Lambda)$, all with mn entries. This is contrary to the aims of never having to compute or store a matrix of full dimension $m \times n$ at any intermediate step of the integration.

If Ω_1 or Ω_2 are not (Fourier-)diagonalizable, the computation of the matrix exponential is realized by iterative methods, cf. Section 5.2.2. However, also in this case a (storage) efficient implementation is not possible, since then the products arising in the update steps of the projector-splitting integrator cannot be computed without computing $A(t)$ and $B(t)$ explicitly. Recall, that the product $A(\tau)\mathbf{E}$ for some $\mathbf{E} \in \mathbb{C}^{n\times r}$ is essential in these steps. Since the exact solution is only available in vectorized form, we consider this product in vectorized form also. By Theorem A.14, we then have

$$\operatorname{vec}(A(\tau)\mathbf{E}) = (\mathbf{E}^T \otimes I_m)\operatorname{vec} A(\tau) = (\mathbf{E}^T \otimes I_m)\big(\cos(\tau\Omega)\operatorname{vec}\mathbf{A}_0 + \tau\operatorname{sinc}(\tau\Omega)\operatorname{vec}\mathbf{B}_0\big),$$

or equivalently

$$A(\tau)\mathbf{E} = \operatorname{vec}^{-1}_{m,n}\big(\cos(\tau\Omega)\operatorname{vec}\mathbf{A}_0\big)\mathbf{E} + \operatorname{vec}^{-1}_{m,n}\big(t\operatorname{sinc}(\tau\Omega)\operatorname{vec}\mathbf{B}_0\big)\mathbf{E},$$

where $\operatorname{vec}^{-1}_{m,n}$ denotes the devectorization, cf. Definition A.12. Neither of the addends can be computed without computing vectors of length mn, and therefore we cannot compute the product $A(\tau)\mathbf{E}$ storage efficiently. The same holds for the product $A(\tau)^H\mathbf{E}$ for $\mathbf{E} \in \mathbb{C}^{m\times r}$, and therefore none of the update steps of the projector-splitting integrator can be performed storage-economical. These difficulties motivate the splitting as in (4.50) for the construction of the `stlostiff` scheme.

CHAPTER 6

Rank-adaptivity

For all dynamical low-rank integrators discussed in the previous chapters it is required to fix the approximation rank at the beginning of the integration. It is important to choose it carefully: If the approximation rank is too small, the low-rank approximation lacks accuracy since the low-rank error dominates the time-discretization error. Conversely, if the rank is chosen too large, the algorithms become inefficient. However, in many applications the appropriate choice of this rank is not known a priori, and it might also vary over time.

Several strategies of rank-adaptivity for dynamical low-rank integrators have been proposed in the past. E.g., in [Dektor et al., 2021], rank-adaptivity for tensor methods for high-dimensional PDEs was based on a functional tensor train series expansion. For finite-dimensional parametrized Hamiltonian systems modeling non-dissipative phenomena, a rank-adaptive structure-preserving reduced basis method was introduced in [Hesthaven et al., 2022]. In [Ceruti et al., 2022] a rank-adaptive strategy for the unconventional robust integrator (cf. Section 3.4.2) was derived. It makes use of unique properties of this specific scheme and cannot be generalized to other dynamical low-rank integrators.

We introduce a general strategy for choosing the rank adaptively, which is applicable for all dynamical low-rank integrators discussed in this thesis. The ansatz has been submitted for publication in [Hochbruck et al., 2022b]. It is designed such that the only input parameter is the step size of the underlying splitting method, i.e., the Lie-Trotter splitting for first-order and the Strang splitting for second-order matrix differential equations. The aim is to ensure that the low-rank error of the adaptive dynamical low-rank approximation does not reduce the approximation order of the respective underlying splitting scheme applied to the full problem. For this, we propagate one additional singular value which is used to accept or reject the current integration step and allows one to choose the rank for the next step. These decisions are made based on an estimator of the local time-discretization error.

In the following, we first present a strategy to select the appropriate rank adaptively based on a given tolerance threshold and on the singular values of the numerically computed approximation. We also show how to switch between low-rank manifolds of distinct rank. The corresponding technique was already proposed in [Lubich and Oseledets, 2014] and recently also in [Hu and Wang, 2021]. Afterwards, we propose a heuristic for choosing the tolerance threshold and derive a computable criterion by combination of estimators for time-discretization and low-rank errors. Also, we comment shortly on the several rank-adaptive schemes derived from this ansatz and highlight specific properties.

6.1 Selecting the rank

The main idea of our ansatz in rank-adaptivity is to approximate the solution to (3.1) or (4.1) by a matrix of rank r_k in the kth time step, but to propagate a solution with one additional singular value. I.e., the low-rank approximations are computed in the manifold $\mathcal{M}_{r_k+1}$. The supplementary information available by the added singular value is used as an indicator whether the rank for the current or next time step has to be adjusted. For simplicity, we first restrict ourselves to the projector-splitting integrator. The extension to the other dynamical low-rank integrators is provided in Section 6.3.

Given the factors $\mathbf{U}_k, \mathbf{S}_k, \mathbf{V}_k$ of a low-rank approximation $\mathbf{A}_k = \mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^H \in \mathcal{M}_{r_k+1}$, $\mathbf{A}_k \approx A(t_k)$, one step of the projector-splitting integrator computes the factors $\mathbf{U}_{k+1}, \mathbf{S}_{k+1}, \mathbf{V}_{k+1}$ of the approximation $\mathbf{A}_{k+1} \approx A(t_k)$. Due to the orthogonality of $\mathbf{U}_{k+1}$ and $\mathbf{V}_{k+1}$, the singular values of $\mathbf{A}_{k+1}$ are the singular values of the small matrix $\mathbf{S}_{k+1}$. We compute the singular value decomposition of $\mathbf{S}_{k+1}$,

$$\mathbf{S}_{k+1} = \mathbf{P}_{k+1}\widehat{\mathbf{\Sigma}}_{k+1}\mathbf{Q}_{k+1}^H, \qquad \widehat{\mathbf{\Sigma}}_{k+1} = \operatorname{diag}(\widehat{\boldsymbol{\sigma}}_1^{(k+1)}, \dots, \widehat{\boldsymbol{\sigma}}_{r_k}^{(k+1)}, \widehat{\boldsymbol{\sigma}}_{r_k+1}^{(k+1)}),$$

where $\mathbf{P}_{k+1}, \mathbf{Q}_{k+1} \in \mathcal{V}_{r_k+1,r_k+1}$ and $\widehat{\boldsymbol{\sigma}}_1^{(k+1)} \geq \dots \geq \widehat{\boldsymbol{\sigma}}_{r_k}^{(k+1)} \geq \widehat{\boldsymbol{\sigma}}_{r_k+1}^{(k+1)} \geq 0$. The reduced singular value decomposition of $\mathbf{A}_{k+1}$ then reads

$$\mathbf{A}_{k+1} = (\mathbf{U}_{k+1}\mathbf{P}_{k+1})\widehat{\mathbf{\Sigma}}_{k+1}(\mathbf{V}_{k+1}\mathbf{Q}_{k+1})^H.$$

Given a tolerance $\texttt{tol}_{k+1}$, we determine the rank $r^\star = r_{k+1}$ such that

$$\widehat{\boldsymbol{\sigma}}_{r^\star+1}^{(k+1)} < \texttt{tol}_{k+1} \leq \widehat{\boldsymbol{\sigma}}_{r^\star}^{(k+1)}. \tag{6.1}$$

We distinguish three cases:

1. *Augmentation case:* If $\widehat{\boldsymbol{\sigma}}_{r_k+1}^{(k+1)} \geq \texttt{tol}_{k+1}$, the step is rejected. Defining $r_k^\star = r_k + 1$, the step is recalculated with rank $r_k^\star + 1$. Hence, we need low-rank factors with column dimension $r_k^\star + 1$, which is not true for the factors $\mathbf{U}_k, \mathbf{S}_k$, and $\mathbf{V}_k$. We therefore append a zero entry to $\mathbf{S}_k$,

$$\mathbf{S}_k^\star = \begin{bmatrix} \mathbf{S}_k & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{C}^{(r_k^\star+1)\times(r_k^\star+1)}. \tag{6.2a}$$

The matrices $\mathbf{U}_k$ and $\mathbf{V}_k$ are augmented by appending unit vectors $u \in \mathbb{C}^m$ and $v \in \mathbb{C}^n$ such that

$$\mathbf{U}_k^\star = \begin{bmatrix} \mathbf{U}_k & u \end{bmatrix} \in \mathcal{V}_{m,r_k^\star+1}, \qquad \mathbf{V}_k^\star = \begin{bmatrix} \mathbf{V}_k & v \end{bmatrix} \in \mathcal{V}_{n,r_k^\star+1}. \tag{6.2b}$$

Numerical tests indicate, that choosing u and v as random vectors and orthonormalizing them against $\mathbf{U}_k$ and $\mathbf{V}_k$ is reliable and robust. Clearly, $\mathbf{U}_k^\star\mathbf{S}_k^\star(\mathbf{V}_k^\star)^H = \mathbf{U}_k\mathbf{S}_k\mathbf{V}_k^H = \mathbf{A}_k$, i.e., the initial value of the current step has not changed. However, the numerical approximation is now able to evolve to rank $r_k^\star + 1$. Starting from the new factorization of $\mathbf{A}_k$ in terms of the matrices

$\mathbf{U}_k^\star, \mathbf{S}_k^\star$, and $\mathbf{V}_k^\star$, the $(k+1)$-st integration step is recomputed, yielding the approximation $\mathbf{A}_{k+1}^\star = \mathbf{U}_{k+1}^\star \mathbf{S}_{k+1}^\star (\mathbf{V}_{k+1}^\star)^H$. This procedure is repeated until

$$\widehat{\boldsymbol{\sigma}}_{r_k^\star+1}^{(k+1)} < \texttt{tol}_{k+1} \leq \widehat{\boldsymbol{\sigma}}_{r_k^\star}^{(k+1)}$$

is satisfied, see also Algorithm 12. Then, the step is finally accepted and the rank for the next step is chosen as $r^\star = r_k^\star$. The initial values for the next step are

$$\mathbf{U}_{k+1} = \mathbf{U}_{k+1}^\star, \qquad \mathbf{S}_{k+1} = \mathbf{S}_{k+1}^\star, \qquad \mathbf{V}_{k+1} = \mathbf{V}_{k+1}^\star,$$

all of column dimension $r_k^\star + 1$.

2. *Reduction case:* If $\widehat{\boldsymbol{\sigma}}_{r_k}^{(k+1)} < \texttt{tol}_{k+1}$, an accurate approximation is also available with a smaller rank. The step is accepted, but the rank for the next step is determined by

$$r^\star = \max\left\{ \operatorname{argmin}\{j \mid \widehat{\boldsymbol{\sigma}}_j^{(k+1)} < \texttt{tol}_{k+1}\},\ r_k - 2\right\},$$

i.e., the rank is either reduced by 1 or 2. Hence, the rank decays slowly, which prevents sudden drops of the rank. As initial values for the next step, one chooses

$$\mathbf{S}_{k+1} = \widetilde{I}^T \Sigma_{k+1} \widetilde{I}, \qquad \mathbf{U}_{k+1} = (\mathbf{U}_{k+1}\mathbf{P}_{k+1})\widetilde{I}, \qquad \mathbf{V}_{k+1} = (\mathbf{V}_{k+1}\mathbf{Q}_{k+1})\widetilde{I},$$

where

$$\widetilde{I} = \begin{bmatrix} I_{r_{k+1}+1} \\ 0 \end{bmatrix} \in \mathbb{R}^{(r_k+1)\times(r_{k+1}+1)}.$$

In order to prevent rank-oscillations, rank-reduction is prohibited within the first 10 steps after an augmentation step.

3. *Persistent case:* If (6.1) is satisfied, the step is accepted and $r^\star = r_k$.

6.2 Choice of tolerance

It remains to choose a tolerance threshold $\texttt{tol}_k$, $k = 0, 1, \dots$ in (6.1). The possibly easiest choice is to employ an absolute tolerance threshold which is independent of the iteration index k,

$$\texttt{tol}_k = \texttt{abstol}, \qquad k = 0, 1, \dots.$$

Then all singular values which are smaller than `abstol` are discarded in the low-rank approximation. However, the trajectories of the singular values are ignored. This might cause issues, especially if the largest singular value becomes smaller over time. If it falls below `abstol`, all singular values are discarded.

A straightforward adjustment is to either require $r \geq 1$ or to introduce a fixed relative tolerance `reltol` and to define

$$\texttt{tol}_k = \widehat{\boldsymbol{\sigma}}_1^{(k)} \cdot \texttt{reltol}, \qquad k = 0, 1, \dots, \tag{6.3}$$

where $\widehat{\boldsymbol{\sigma}}_1^{(k)}$ denotes the largest singular value of the numerical approximation $\mathbf{A}_k$. Now the trajectory of the largest singular value is taken into account. Even if the singular value drops significantly over time, the low-rank approximation is always at least of rank 1.

Algorithm 12: Augmentation

```
augmentation($\mathbf{U}, \mathbf{S}, \mathbf{V}, \Delta A, r$, tol)
Input  : factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of rank-$(r+1)$ approximation $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H$ with $\mathbf{U} \in \mathcal{V}_{m,r+1}$,
         $\mathbf{V} \in \mathcal{V}_{n,r+1}$, $\mathbf{S} \in \mathbb{C}^{(r+1)\times(r+1)}$, functions for products with $\Delta A$, tolerance tol

ready = False
while not ready do
    $r = r + 1$
    choose $u \in \mathbb{C}^m$ orthonormal to $\mathbf{U}$ (e.g., random)
    choose $v \in \mathbb{C}^n$ orthonormal to $\mathbf{V}$ (e.g., random)
    compute $\mathbf{U} = \mathbf{U}^\star$, $\mathbf{S} = \mathbf{S}^\star$, $\mathbf{V} = \mathbf{V}^\star$ as in (6.2)
    $\mathbf{U}, \mathbf{S}, \mathbf{V}, \mathbf{L}$ = prsi($\mathbf{U}, \mathbf{S}, \mathbf{V}, r+1, \Delta A$)
    compute singular values $\widehat{\boldsymbol{\sigma}}_1, \ldots, \widehat{\boldsymbol{\sigma}}_{r+1}$ of $\mathbf{S}$
    ready = ($\widehat{\boldsymbol{\sigma}}_{r+1} <$ tol)
end
Return $\mathbf{U}, \mathbf{S}, \mathbf{V}, \mathbf{L}, r$
Output: factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of rank-$(r+1)$ approximation to $\mathbf{A} + \Delta A$ and $\mathbf{L} = \mathbf{V}\mathbf{S}^H$ with
         $\mathbf{U} \in \mathcal{V}_{m,r+1}$, $\mathbf{V} \in \mathcal{V}_{n,r+1}$, $\mathbf{S} \in \mathbb{C}^{(r+1)\times(r+1)}$
```

A major drawback of the ansatz in (6.3) is that a suitable choice of `reltol` is in general not known beforehand. Obviously, the smaller `reltol`, the more singular values can be taken into account. This enlarges the approximation quality, but also increases the computational effort. Larger values of `reltol` reduce the effort, but a good approximation may not be guaranteed anymore.

We propose a different ansatz for choosing `tol`. Instead of linking the tolerance threshold to the trajectory of the largest singular value, we take the time-discretization error into account. This is motivated by the results in Theorem 3.6 for the projector-splitting integrator and Theorem 4.11 for the St-LO scheme. There it was shown that for both low-rank schemes the global error splits into a low-rank error and a time-discretization error contribution. Clearly, if the time-discretization error is already large, the approximation rank can be chosen smaller, maintaining the convergence order. Contrary, a small time-discretization error might be spoiled by a large low-rank approximation error, thus a larger approximation rank should be chosen. We therefore suggest to construct rank-adaptive dynamical low-rank integrators based on balancing the low-rank approximation and the time-discretization errors such that the convergence order (and the time-discretization error) is not impaired by the low-rank error.

The error analysis of the projector-splitting integrator provided in [Kieri et al., 2016], see also Theorem 3.6, shows exponential growth of the error w.r.t. the final time T. For the St-LO scheme we have seen a similar behavior, cf. Theorem 4.11. However, numerical experiments indicate that this is a pessimistic bound, and the errors grow much slower in T. Since the exact behavior of the global error w.r.t. T is in general not known, we use an estimator for the time-discretization error, which is the basis for computing the tolerance threshold `tol`.

6.2.1 Time-discretization error estimation via Richardson extrapolation

In our implementation, we approximate the evolution of the error by monotonically increasing piecewise linear functions. The respective slopes are recalculated every M steps, where $M \in \mathbb{N}$ is suitably chosen. The practical estimation of the time-discretization error is done as follows:

Starting from an approximation $\mathbf{A}_{\ell M} \approx A(t_{\ell M})$, we compute an approximation to $A(t_{\ell M+1})$ with the rank-adaptive integrator and step size τ. In this step, we prevent rank-reduction. The (propagated) numerical approximation $\mathbf{A}_{\ell M+1}$ possesses rank $r^* = r_{\ell M+1} + 1$. The subsequent time step is now performed with rank r^*, which yields the approximation $\mathbf{A}_{\ell M+2}$. In parallel, we perform four steps with step size $\frac{\tau}{2}$ and approximation rank r^*, started from $\mathbf{A}_{\ell M}$. By this, we obtain alternative (and in general more accurate) approximations $\breve{\mathbf{A}}_{\ell M+1} \approx A(t_{\ell M+1})$ and $\breve{\mathbf{A}}_{\ell M+2} \approx A(t_{\ell M+2})$. If the chosen method converges with order $p \in \mathbb{N}$ in time, Richardson extrapolation [Zlatev et al., 2017] allows one to estimate the propagated errors of $\mathbf{A}_{\ell M+1}$ and $\mathbf{A}_{\ell M+2}$ as

$$\|A(t_{\ell M+1}) - \mathbf{A}_{\ell M+1}\| \approx \frac{2^p}{2^p - 1}\|\mathbf{A}_{\ell M+1} - \breve{\mathbf{A}}_{\ell M+1}\| =: \mathtt{err}_\ell^{\mathrm{I}},$$

and

$$\|A(t_{\ell M+2}) - \mathbf{A}_{\ell M+2}\| \approx \frac{2^p}{2^p - 1}\|\mathbf{A}_{\ell M+2} - \breve{\mathbf{A}}_{\ell M+2}\| =: \mathtt{err}_\ell^{\mathrm{II}},$$

cf. [Constantinescu, 2018, Section 5]. We let

$$\zeta_\ell = \frac{\mathtt{err}_\ell^{\mathrm{II}}}{2\mathtt{err}_\ell^{\mathrm{I}}},$$

and define the quantity $\mathtt{err}_\ell$ recursively via

$$\mathtt{err}_{\ell+1} = \mathtt{err}_\ell + M\zeta_\ell \mathtt{err}_\ell^{\mathrm{I}}, \qquad \ell = 0, 1, \dots, \qquad \mathtt{err}_0 = 0.$$

The estimation of the propagated error of $\mathbf{A}_k \approx A(t_k)$ for $k = \ell M + j$ then reads

$$\|A(t_{\ell M+j}) - \mathbf{A}_{\ell M+j}\| \approx \mathtt{err}_\ell + j\zeta_\ell \mathtt{err}_\ell^{\mathrm{I}} =: \mathtt{tde}_k, \qquad j = 1, 2, \dots, M. \tag{6.4}$$

Remark 6.1. Computing the norm of the difference of the low-rank matrices $\mathbf{A}$ and $\breve{\mathbf{A}}$ should not involve the matrices themselves, but their low-rank factors. By the rules of the Frobenius norm and inner product (cf. Theorem A.7), it is

$$\begin{aligned}\|\mathbf{A} - \breve{\mathbf{A}}\|^2 &= \|\mathbf{U}\mathbf{S}\mathbf{V}^H - \breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H\|^2 \\ &= \|\mathbf{U}\mathbf{S}\mathbf{V}^H\|^2 - 2\,\mathrm{Re}\langle \mathbf{U}\mathbf{S}\mathbf{V}^H, \breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H\rangle + \|\breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H\|^2 \\ &= \|\mathbf{S}\|^2 + \|\breve{\mathbf{S}}\|^2 - 2\,\mathrm{Re}\,\mathrm{tr}(\mathbf{V}\mathbf{S}^H\mathbf{U}^H\breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H) \\ &= \|\mathbf{S}\|^2 + \|\breve{\mathbf{S}}\|^2 - 2\,\mathrm{Re}\,\mathrm{tr}\left((\breve{\mathbf{V}}^H\mathbf{V})\mathbf{S}^H(\mathbf{U}^H\breve{\mathbf{U}})\breve{\mathbf{S}}\right),\end{aligned}$$

where the last equality holds by the identity $\mathrm{tr}(AB) = \mathrm{tr}(BA)$. This representation allows us to compute the norm of the distance between $\mathbf{A}$ and $\breve{\mathbf{A}}$ without having to compute or store any matrix of full dimension in any intermediate step. However, it was numerically observed that this expression suffers from cancellation. With the orthogonal projector $\Pi_{\mathbf{U}} = \mathbf{U}\mathbf{U}^H$ onto the column space of $\mathbf{U}$ and the orthogonal projector $\Pi_{\mathbf{U}}^{\perp} = I_m - \Pi_{\mathbf{U}}$ onto its orthogonal complement, see (2.20), we rewrite $\breve{\mathbf{A}}$ as the sum

$$\breve{\mathbf{A}} = \Pi_{\mathbf{U}}\breve{\mathbf{A}} + \Pi_{\mathbf{U}}^{\perp}\breve{\mathbf{A}}.$$

This yields

$$\begin{aligned}\|\mathbf{A} - \breve{\mathbf{A}}\|^2 &= \|\mathbf{U}\mathbf{S}\mathbf{V}^H - \Pi_{\mathbf{U}}\breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H - \Pi_{\mathbf{U}}^{\perp}\breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H\|^2 \\ &= \|\mathbf{U}(\mathbf{S}\mathbf{V}^H - \mathbf{U}^H\breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H)\|^2 + \|\Pi_{\mathbf{U}}^{\perp}\breve{\mathbf{U}}\breve{\mathbf{S}}\breve{\mathbf{V}}^H\|^2 \\ &= \|\mathbf{S}\mathbf{V}^H - (\mathbf{U}^H\breve{\mathbf{U}})\breve{\mathbf{S}}\breve{\mathbf{V}}^H\|^2 + \|\breve{\mathbf{U}}\breve{\mathbf{S}} - \mathbf{U}(\mathbf{U}^H\breve{\mathbf{U}})\breve{\mathbf{S}}\|^2,\end{aligned}$$

which we observed to be more robust in our examples. ◇

6.2.2 Low-rank error estimation

For estimating the low-rank approximation error, consider first the exact solution A at time t_{k+1} and denote its singular values by $\sigma_1 \geq \ldots \geq \sigma_n \geq 0$. The rank-$r_{k+1}$ best-approximation $\mathbf{A}_{k+1}^{\text{best}}$ to $A(t_{k+1})$ fulfills

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

so that

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

The bound in the numerator is a *worst case* estimate, where all singular values $\sigma_{r_{k+1}+1}, \ldots, \sigma_n$ are of the same size. In practice, this is often not the case, especially if A is well-approximated by a low-rank matrix. Neither the singular values of the exact solution are known, nor the best-approximation $\mathbf{A}_{k+1}^{\text{best}}$. For a computable bound, we therefore replace the singular values σ_i of $A(t_{k+1})$ by the singular values $\widehat{\boldsymbol{\sigma}}_i^{(k+1)}$ of the low-rank approximation $\mathbf{A}_{k+1}$, and the best-approximation $\mathbf{A}_{k+1}^{\text{best}}$ by $\mathbf{A}_{k+1}$. The unknown ratio between the norms of the exact solution and the low-rank approximation is estimated as 1. Altogether, this gives

$$\|A(t_{k+1}) - \mathbf{A}_{k+1}\| \lesssim \widehat{\boldsymbol{\sigma}}_{r_{k+1}+1}^{(k+1)}\sqrt{n - r_{k+1}}.$$

6.2.3 Tolerance threshold

The combination of the estimators of time-discretization and low-rank approximation error finally yields a computable threshold `tol`. Enforcing the low-rank error to fall below the time-discretization error yields the inequality

$$\widehat{\boldsymbol{\sigma}}_{r_{k+1}+1}^{(k+1)}\sqrt{n - r_{k+1}} \leq \texttt{tde}_{k+1},$$

where $\texttt{tde}_{k+1}$ is given in (6.4). Solving for $\widehat{\boldsymbol{\sigma}}_{r_{k+1}+1}^{(k+1)}$ yields the condition

$$\widehat{\boldsymbol{\sigma}}_{r_{k+1}+1}^{(k+1)} \leq \frac{\texttt{tde}_{k+1}}{\sqrt{n - r_{k+1}}} = \frac{\texttt{err}_\ell + j\zeta_\ell\texttt{err}_\ell^{\text{I}}}{\sqrt{n - r_{k+1}}} =: \texttt{tol}_{k+1}, \qquad k + 1 = \ell M + j. \tag{6.5}$$

Though these heuristics work well in our numerical experiments, cf. Chapter 7, they are only reliable if the low-rank approximation error is small compared to the time-discretization error right from the start of the integration. Therefore it is required to determine a suitable initial rank r_0.

In our implementation, we use the following heuristic: We start the iteration from a low-rank approximation to A_0 with rank $r_0 = 5$ and perform ν steps (with ν small, e.g., $\nu = 5$). In this phase, we prohibit rank reduction. Let r^* denote the number of singular values of $\mathbf{A}_\nu \approx A(t_\nu)$ which are greater or equal to $\texttt{tol}_\nu$ given in (6.5). If $r^* < r_0$, we continue the integration with $r_{\nu+1} = r^*$. Otherwise, we rerun the initializing process for r_0 multiplied by 2, until $r^* < r_0$ holds.

Remark 6.2. By design, if $r^* \gtrsim r_0$, i.e., r^* is just slightly larger than r_0, the rank r_0 is doubled and might cause a significant overestimation of the initial rank for the first ν steps. The favorable property of the dynamical low-rank integrators to be robust with respect to the presence of small singular values, which is inherited from the projector-splitting integrator, ensures that the approximation quality does not deteriorate even in this situation. Also, since we overestimate the rank only in the first ν steps, the computational overhead is small. ◇

6.3 Rank-adaptive algorithms

The rank-adaptive version of the projector-splitting integrator Algorithm 2 is called `raprsi` for **r***ank*-**a***daptive* **pr***ojector*-**s***plitting* **i***ntegrator* in the following. A single step of the `raprsi` scheme is given in Algorithm 13.

Algorithm 13: Rank-adaptive projector-splitting integrator, single step

```
1  raprsi($\mathbf{U}, \mathbf{S}, \mathbf{V}, r, \Delta A, p$)
   Input  : factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of rank-$(r+1)$ approximation $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^H \approx A(t)$ with $\mathbf{U} \in \mathcal{V}_{m,r+1}$,
            $\mathbf{V} \in \mathcal{V}_{n,r+1}$, $\mathbf{S} \in \mathbb{C}^{(r+1)\times(r+1)}$, functions for products with $\Delta A$
2
3  $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1, \mathbf{L}_1 = \texttt{prsi}\big(\mathbf{U}, \mathbf{S}, \mathbf{V}, r+1, \Delta A\big)$
4  compute SVD $\mathbf{S}_1 = \mathbf{P}\widehat{\mathbf{\Sigma}}\mathbf{Q}^H$ where $\widehat{\mathbf{\Sigma}} = \operatorname{diag}(\widehat{\boldsymbol{\sigma}}_1, \dots, \widehat{\boldsymbol{\sigma}}_{r+1})$
5  compute tol according to Section 6.2
6  if $\widehat{\boldsymbol{\sigma}}_r <$ tol then
7      $r_1 = \operatorname{argmin}\{j \mid \widehat{\boldsymbol{\sigma}}_{j+1} < \texttt{tol}\}$
8      $\widetilde{I} = \big[I_{r_1+1}\ 0\big]^T \in \mathbb{C}^{(r+1)\times(r_1+1)}$
9      $\mathbf{U}_1 = (\mathbf{U}_1\mathbf{P})\widetilde{I}$
10     $\mathbf{S}_1 = \widetilde{I}^T\widehat{\mathbf{\Sigma}}\widetilde{I}$
11     $\mathbf{V}_1 = (\mathbf{V}_1\mathbf{Q})\widetilde{I}$
12     $\mathbf{L}_1 = \mathbf{V}_1\mathbf{S}_1^H$
13 else if $\widehat{\boldsymbol{\sigma}}_{r+1} \geq$ tol then
14     $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1, \mathbf{L}_1, r_1 = \texttt{augmentation}(\mathbf{U}, \mathbf{S}, \mathbf{V}, \Delta A, r, \texttt{tol})$
15 Return $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1, \mathbf{L}_1, r_1$
   Output: factors $\mathbf{U}_1, \mathbf{S}_1, \mathbf{V}_1$ of rank-$(r_1+1)$ approximation $\mathbf{A}_1 = \mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^H \approx A(t+\tau)$ and
            $\mathbf{L}_1 = \mathbf{V}_1\mathbf{S}_1^H$ with $\mathbf{U}_1 \in \mathcal{V}_{m,r_1+1}$, $\mathbf{V}_1 \in \mathcal{V}_{n,r_1+1}$, $\mathbf{S}_1 \in \mathbb{C}^{(r_1+1)\times(r_1+1)}$
```

The rank-adaptive equivalent of the `prsistiff` scheme (cf. Section 3.3) is named `raprsistiff`. Since the linear subproblem (3.13a) preserves the rank of its initial value, rank-adaptivity is only applied in the integration of the nonlinear subproblem (3.13b). One single time step of the `raprsistiff` is hence obtained by replacing the `prsiF` scheme in line 7 of Algorithm 4 by the `raprsi` method.

Similarly, we derive a rank-adaptive version of the St-LO scheme by replacing the `prsi` routines in lines 2 and 3 in Algorithm 7 by the `raprsi` scheme. This new integrator is named **r***ank-***a***daptive* **St-LO** (`rastlo`) scheme. A rank-adaptive version of the `stlovar` scheme is derived in the same way and is named `rastlovar`.

With the same idea we obtain the `rastlotangent` scheme as rank-adaptive counterpart to the `stlotangent` method given in Algorithm 10. Note that for this integrator rank-adaptivity is only used in the **A**-step of the integration (line 3), since the approximation rank $r_{\mathbf{A}}$ implicitly determines the column dimension of the low-rank factors of the approximation $\mathbf{B} \approx A'$, see also (4.63).

For stiff second-order matrix differential equations (4.49), we equip the integrator `stlostiff` displayed in (4.53) with the adaptivity schemes described above. For the sake of efficiency, the rank is only allowed to change in the integration of the nonlinear subproblem, though the linear subproblems in general do not preserve the rank. Only if rank-augmentation needs to be performed, the affected substeps of Algorithm 8 are recomputed. The resulting rank-adaptive scheme is named `rastlostiff`.

Rank-adaptivity for the St-LO scheme

In contrast to all other dynamical low-rank integrators discussed within this thesis, the St-LO scheme computes approximations to A and A' on a staggered grid. This causes a problem in the computation of the tolerance threshold as presented in Section 6.2. Richardson extrapolation expects the local time-discretization error to converge with order $p+1$ if the method converges globally with order p. However, if the approximations are not computed on the same grid, this property is violated, which can already be observed for the leapfrog scheme: Consider the linear problem

$$A''(t) = LA(t), \quad t \in [0,T], \qquad A(0) = A_0, \quad A'(0) = B(0),$$

and compute two sets of approximations to $A(t_2)$ and $A'(t_{\frac{3}{2}})$. The first one is obtained by performing one step of the leapfrog scheme in its staggered formulation (4.4a) with step size τ, started from approximations $A_1 \approx A(t_1)$, $B_{\frac{1}{2}} \approx A'(t_{\frac{1}{2}})$. This yields the approximations $A_2 \approx A(t_2)$ and $B_{\frac{3}{2}} \approx A'(t_{\frac{3}{2}})$. The second set $\breve{A}_2, \breve{B}_{\frac{3}{2}}$ results from performing two steps of the scheme (4.4a) with step size $\frac{\tau}{2}$, also started from $A_1, B_{\frac{1}{2}}$. Straightforward calculation yields the expressions

$$\begin{aligned} B_{\frac{3}{2}} &= B_{\frac{1}{2}} + \tau L A_1, \\ A_2 &= A_1 + \tau B_{\frac{1}{2}} + \tau^2 L A_1, \end{aligned}$$

as well as

$$\begin{aligned} \breve{B}_{\frac{3}{2}} &= B_{\frac{1}{2}} + \tau L A_1 + \frac{\tau^2}{4} L B_{\frac{1}{2}} + \frac{\tau^3}{8} L^2 A_1, \\ \breve{A}_1 &= A_1 + \tau B_{\frac{1}{2}} + \frac{3\tau^2}{4} L A_1 + \frac{\tau^3}{8} L B_{\frac{1}{2}} + \frac{\tau^4}{16} L^2 A_1. \end{aligned}$$

Comparison of $B_{\frac{3}{2}}$ and $\breve{B}_{\frac{3}{2}}$ shows that the error between these quantities is of order $\mathcal{O}(\tau^2)$, and the same holds for the error between A_2 and $\breve{A}_2$. Thus, also for general right-hand sides $F(A)$ we cannot expect Richardson extrapolation to yield proper results if the approximations are not computed on the same grid.

A remedy for this issue is to modify the `rastlo` routine. After performing ℓM steps, one is given the approximations $\mathbf{B}_{\ell M-\frac{1}{2}}$ and $\mathbf{A}_{\ell M}$. Computing a subsequent **B**-step with step size $\frac{\tau}{2}$ yields $\mathbf{B}_{\ell M} \approx A'(t_{\ell M})$. Since now the approximations to A and its derivative are given on the same grid, the estimation of the time-discretization error as explained in Section 6.2.1 is reliable. For this estimation however the St-LO scheme needs to be replaced by the `rastlovar` scheme in order to remain on a non-staggered time grid. This gives the approximations $\mathbf{B}_{\ell M+2}$ and $\mathbf{A}_{\ell M+2}$. For proceeding the simulation, the next **B**-step is performed again with halved step size $\frac{\tau}{2}$, and all further **A**- and **B**-steps are carried out with step size τ until one reaches the approximations $\mathbf{B}_{(\ell+1)M-\frac{1}{2}}$ and $\mathbf{A}_{(\ell+1)M}$. Then one repeats the whole procedure.

CHAPTER 7

Numerical experiments

We conclude this thesis with numerical experiments for matrix differential equations resulting from space discretizations of PDEs. They illustrate the performance of the constructed fixed-rank and rank-adaptive dynamical low-rank integrators for first-order and second-order matrix differential equations.

All considered PDEs are imposed on a rectangular domain

$$\Omega = [-L_x, L_x] \times [-L_y, L_y] \subset \mathbb{R}^2. \tag{7.1}$$

For the discretization in space, we use a uniform mesh with n grid points in x- and m grid points in y-direction, respectively. We denote with

$$h_x = \frac{2L_x}{n}, \quad h_y = \frac{2L_y}{m}, \qquad m, n \in \mathbb{N},$$

the mesh size in x- and y-direction, respectively. The mesh Ω_h depends on the chosen boundary conditions:

1. *Dirichlet boundary conditions*

$$\Omega_h = \{(x_j, y_i) \mid x_j = -L_x + jh_x,\ y_i = -L_y + ih_y,\ 1 \le j \le n-1,\ 1 \le i \le m-1\}, \tag{7.2}$$

2. *periodic boundary conditions*

$$\Omega_h = \{(x_j, y_i) \mid x_j = -L_x + jh_x,\ y_i = -L_y + ih_y,\ 1 \le j \le n,\ 1 \le i \le m\},$$

3. *Neumann boundary conditions*

$$\Omega_h = \{(x_j, y_i) \mid x_j = -L_x + jh_x,\ y_i = -L_y + ih_y,\ 0 \le j \le n,\ 0 \le i \le m\}. \tag{7.3}$$

Errors of the low-rank approximations are measured w.r.t. numerically computed reference solutions, unless the exact solution of the respective problem is known. Since we are only interested in the time-discretization error, reference solution and low-rank solutions are computed on the same spatial grid. Moreover, we always compute relative global errors at $t_k = T$,

$$\texttt{err} = \frac{\|A_k - \mathbf{A}_k\|}{\|A_k\|},$$

between the reference solution A and the low-rank approximation $\mathbf{A}$, as these approximate the relative discrete L^2-norm of the respective functions.

For the rank-adaptive dynamical low-rank integrators, the computation of the tolerance threshold as presented in Section 6.2.1 is performed with $M = 100$, i.e., every 100 steps we perform four additional steps with step size $\frac{\tau}{2}$. This increases the computational effort by 4%. Choosing smaller values for M improves the estimation of the time-discretization error, but also increases the computational cost. In one of our experiments, we found that $M = 10$ was necessary.

All algorithms have been implemented in Python. The codes for recreating the figures in this chapter are available from [Schrammer, 2022].

7.1 Stiff first-order matrix differential equations

In this section, we consider stiff first-order matrix differential equations of the form (3.12). For such problems, a dynamical low-rank integrator has already been proposed in [Ostermann et al., 2019], see also the `prsistiff` scheme (cf. Algorithm 4) in Section 3.3. The purpose of this section is to test the rank-adaptive variant of the `prsistiff` method as it has been designed in Section 6.3.

7.1.1 Nonlinear fractional Ginzburg–Landau equation

The cubic Ginzburg–Landau equation is used to describe a variety of physical phenomena, e.g., superconductivity, superfluidity, and others, cf. [Aranson and Kramer, 2002]. In [Tarasov and Zaslavsky, 2005], the authors derived a fractional generalization of the Ginzburg–Landau equation. Discretization in space by the second-order fractional centered difference method proposed in [Çelik and Duman, 2012] and homogeneous Dirichlet boundary conditions yields the stiff semilinear first-order matrix differential equation

$$A'(t) = -D_y A(t) - A(t)D_x - (\kappa + \mathrm{i}\xi)|A(t)|^2 A(t) + \gamma A(t), \quad t \in [0,T], \qquad A(0) = A_0, \tag{7.4}$$

where D_x and D_y are symmetric Toeplitz matrices with first columns

$$\frac{\nu + \mathrm{i}\eta}{h_x^\alpha}\left[g_0^\alpha, g_1^\alpha, \dots, g_{n-2}^\alpha\right]^T \qquad \text{and} \qquad \frac{\nu + \mathrm{i}\eta}{h_y^\beta}\left[g_0^\beta, g_1^\beta, \dots, g_{m-2}^\beta\right]^T,$$

cf. Definition A.17. Here, $\mathrm{i} = \sqrt{-1}$, $\nu, \kappa > 0$, $\eta, \xi, \gamma \in \mathbb{R}$, and $1 < \alpha, \beta < 2$ denote given parameters, and

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

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

Remark 7.1. Since $\Gamma(z) \to 0$ for $z \to -\infty$ and $\Gamma(z) \to \infty$ for $z \to \infty$, respectively, the computation of the denominator of g_k^μ is numerically unstable if k becomes large. By using the property

$$\Gamma(z+1) = z\Gamma(z) \qquad \text{for} \quad -z \notin \mathbb{N}$$

for the Gamma function one obtains the alternative representation

$$\Gamma(\mu/2 - k + 1)\Gamma(\mu/2 + k + 1) = \Gamma^2(\mu/2 + 1)\prod_{i=1}^{k} \frac{\mu/2 + i}{\mu/2 - k + i}.$$

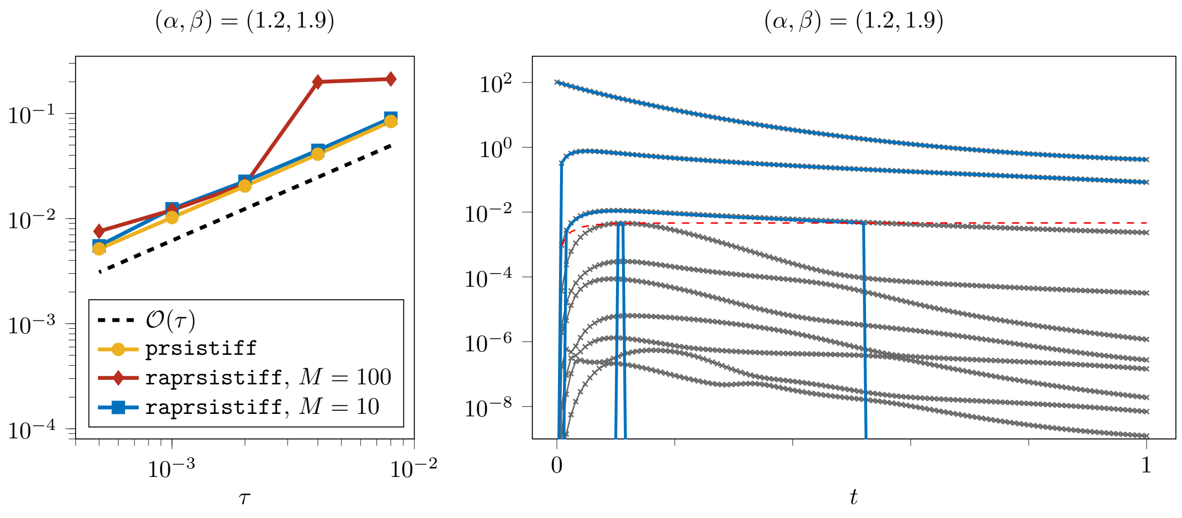

Figure 7.1. Fractional Ginzburg–Landau equation, first experiment with A_0 from (7.5). The left picture shows the relative global error at $T = 1$ for $(\alpha,\beta) = (1.2, 1.9)$, where the fixed-rank approximation (yellow) is computed with $r = 5$. The rank-adaptive approximation was computed with $M = 100$ (orange) and $M = 10$ (blue). The trajectories of the ten largest singular values of the reference solution (gray), the singular values of `raprsistiff` ($M = 10$) for $\tau = 10^{-3}$ (blue), and the computed tolerance threshold (red, dashed) are displayed on the right.

The computation of the denominator based on this representation was observed to be more stable in our experiments. ◇

In [Zhang et al., 2020], (7.4) was solved approximately with the linearized second-order backward differential scheme (LBDF2). A fixed-rank dynamical low-rank integrator for (7.4) based on the considerations from [Ostermann et al., 2019] was proposed in [Zhao et al., 2021]. We compute low-rank approximations to the exact solution of (7.4) with the `prsistiff` and the `raprsistiff` schemes. The solution of the linear subproblem in (7.4) is of the form (3.15) with $L_1 = -D_y$ and $L_2 = -D_x$, respectively. Since D_x and D_y are both Toeplitz matrices, we use the method proposed in [Lee et al., 2010] for computing the action of the matrix exponential, see also Section 5.2.2.

For our tests, we use the parameter sets from [Zhao et al., 2021]. The first one is given as $L_x = L_y = 10$, $m = n = 512$, $\nu = \eta = \kappa = \xi = \gamma = 1$, $T = 1$, $(\alpha,\beta) = (1.2, 1.9)$, with initial value

$$(A_0)_{ij} = 2\,\mathrm{sech}(x_j)\,\mathrm{sech}(y_i)\exp\big(\mathrm{i}(x_j + y_i)\big), \qquad i,j = 1,\ldots,m-1. \tag{7.5}$$

The reference solution is computed with the LBDF2 method and step size $\tau = 10^{-4}$. The relative global errors of the low-rank approximations at time T are displayed in Figure 7.1. For large step sizes, the approximations computed with the `raprsistiff` method show large errors. This is due to an overestimation of the time-discretization error, which causes a tolerance threshold so large that the second largest singular value is discarded. This unfortunate behavior does not appear for $M = 10$.

For the second experiment, the parameters are chosen as $L_x = L_y = 8$, $m = n = 512$, $\nu = \kappa = 1$, $\eta = 0.5$, $\xi = -5$, $\gamma = 3$, $T = 1$, $(\alpha,\beta) = (1.2, 1.9)$, with initial value

$$(A_0)_{ij} = \exp\big(-2(x_j^2 + y_i^2)\big)\exp\big(\mathrm{i}(S_0)_{ji}\big), \quad (S_0)_{ij} = (\mathrm{e}^{x_j+y_i} + \mathrm{e}^{-x_j-y_j})^{-1}, \qquad i,j = 1,\ldots,m-1. \tag{7.6}$$

Figure 7.2 shows the relative global errors at time $T = 1$. The curves for the fixed-rank and rank-adaptive

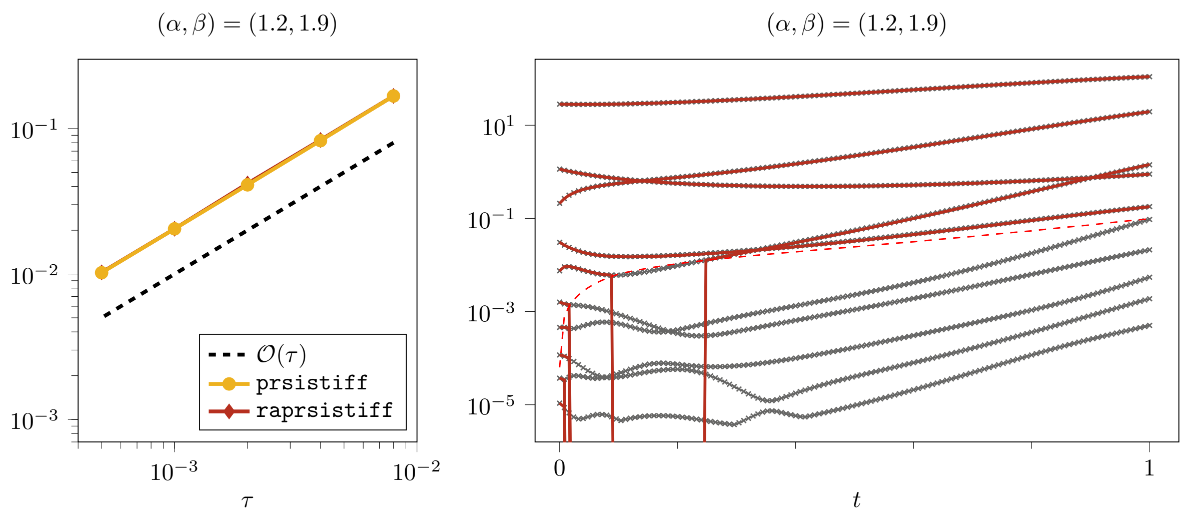

Figure 7.2. Fractional Ginzburg–Landau equation, second experiment with A_0 from (7.6). The left picture shows the relative global error at $T = 1$ for $(\alpha, \beta) = (1.2, 1.9)$, where the fixed-rank approximation is computed with $r = 8$. The trajectories of the ten largest singular values of the reference solution (gray), the singular values of `raprsistiff` for $\tau = 10^{-3}$ (orange), and the computed tolerance threshold (red, dashed) are displayed on the right.

integrators now align almost perfectly.

Similar results for both experiments were observed for the parameters $(\alpha, \beta) = (1.5, 1.5)$,$(1.7, 1.3)$, and $(1.9, 1.2)$ and are available from [Schrammer, 2022].

7.1.2 Nonlinear fractional Schrödinger equation

The nonlinear fractional Schrödinger equation, cf. [Zhao et al., 2014], is a special case of the nonlinear fractional Ginzburg–Landau equation with $\nu = \kappa = \gamma = 0$. In the limit $\alpha, \beta \to 2$ it becomes the classical Schrödinger equation.

For our experiment we take the parameters and the initial value from [Zhao et al., 2014], namely $L_x = L_y = 10$, $n = m = 512$, $\eta = 1$, $\xi = -2$, $T = 0.2$, $(\alpha, \beta) = (1.2, 1.9)$, and

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

We compute low-rank approximations to the exact solution of the problem again with the `prsistiff` and `raprsistiff` methods. A reference solution is computed with the LBDF2 method, using the step size $2 \cdot 10^{-5}$. The results of our experiment are displayed in Figure 7.3. Again, the relative global error curves of the fixed-rank and rank-adaptive schemes match nearly perfectly and clearly indicate convergence of order 1. The results for other choices of α and β are provided in [Schrammer, 2022].

7.2 Second-order matrix differential equations

We proceed with second-order matrix differential equations of the form (4.1). We here use all dynamical low-rank integrators `stlo`, `stlotwostep`, `stlostiff`, `gautschilr`, and `stlotangent` constructed in

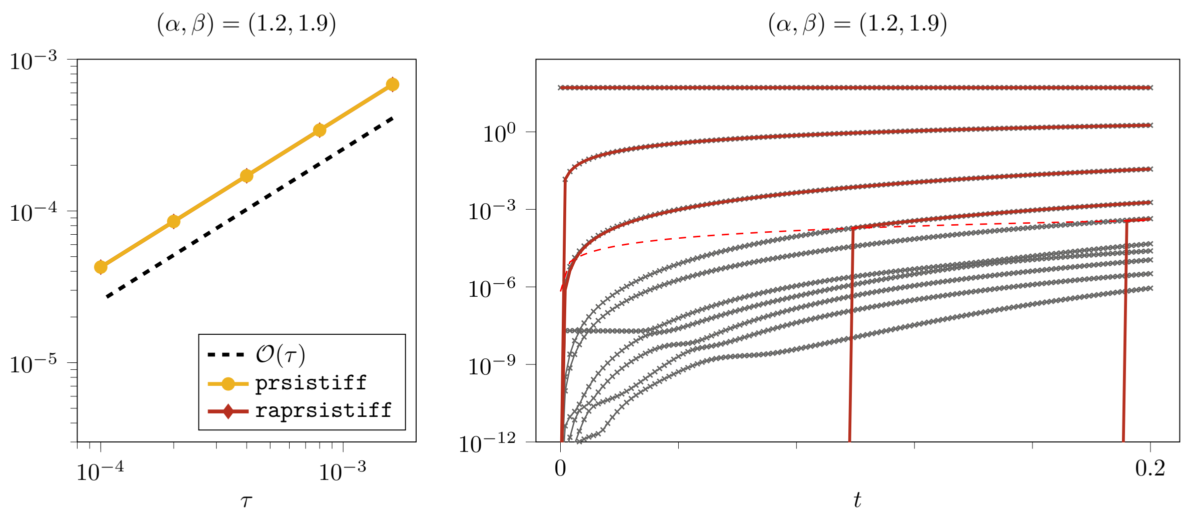

Figure 7.3. Fractional Schrödinger equation. The left picture shows the relative global error at $T = 0.2$ for $(\alpha, \beta) = (1.2, 1.9)$, where the fixed-rank approximation is computed with $r = 5$. The trajectories of the ten largest singular values of the reference solution (gray), the singular values of `raprsistiff` ($M = 100$) for $\tau = 4 \cdot 10^{-4}$ (orange), and the computed tolerance threshold (red, dashed) are displayed on the right.

Chapter 4 to compute low-rank approximations to the exact solutions of (4.1). Moreover, we apply their rank-adaptive versions `rastlo`, `rastlostiff`, and `rastlotangent`.

7.2.1 Homogeneous wave equation

For our first example of a second-order differential equation we consider the homogeneous wave equation

$$\partial_t^2 a(t,x,y) = \Delta a(t,x,y), \qquad t \in [0,T], \quad (x,y) \in \Omega,$$

subject to periodic boundary conditions and suitably chosen initial values

$$a(0,x,y) = \tilde{a}_0(x,y), \quad \partial_t a(0,x,y) = \tilde{b}_0(x,y), \qquad (x,y) \in \Omega.$$

The domain Ω is given in (7.1) with $L_x = 300\pi$ and $L_y = 600\pi$. For the discretization in space we follow the approach in [Schweitzer, 2008, Section 4.1.3] and use fourth order finite differences in x-direction and a pseudospectral method in y-direction. The motivation for this particular discretization is the application in the next experiment in Section 7.2.2. We thus end up with the linear second-order matrix differential equation

$$A''(t) = \mathcal{L}A(t), \qquad A(0) = \widetilde{A}_0, \quad A'(0) = \widetilde{B}_0, \tag{7.7}$$

with

$$(\widetilde{A}_0)_{ij} = 0.12 \exp\Big(-\frac{y_i^2}{l_0^2} - \frac{x_j^2}{w_0^2}\Big), \quad (\widetilde{B}_0)_{ij} = \Big(-\frac{2y_i}{l_0^2}\Big)(\widetilde{A}_0)_{ij}, \qquad i = 1,\dots,m, \quad j = 1,\dots,n.$$

The parameters are chosen as $l_0 = 10\pi$ and $w_0 = 100\pi$. The discrete Laplacian $\mathcal{L}$ acts on $A(t)$ via

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

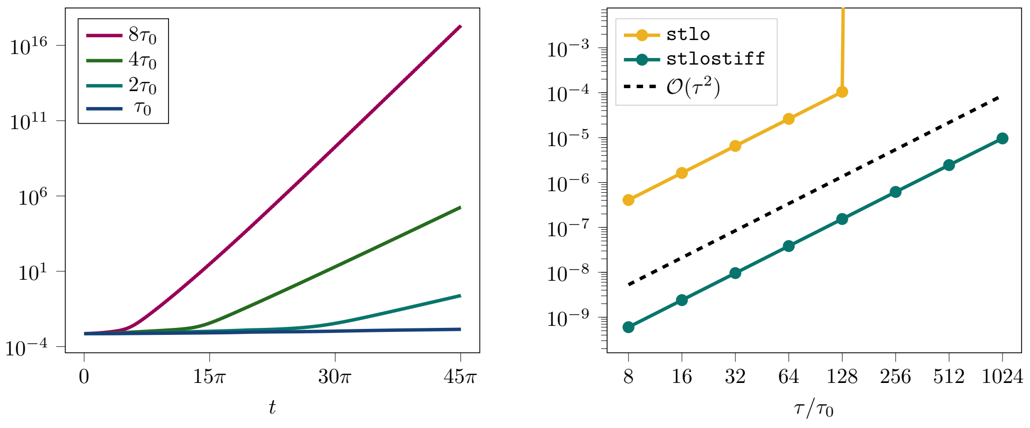

Figure 7.4. Homogeneous wave equation. **Left:** Relative global error in $\mathbf{A}$ between exact solution of (7.7) and low-rank approximations computed with the projector-splitting integrator for $t \in [0, 45\pi]$ and different step sizes, shown from the first time step. **Right:** Relative global error in $\mathbf{A}$ between exact solution of (7.7) and low-rank approximations at $T = 45\pi$. The `stlostiff` scheme allows for larger step sizes compared to the `stlo` scheme.

where $D_x \in \mathbb{R}^{n\times n}$ denotes the symmetric Toeplitz matrix (cf. Definition A.17) with first column

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

and $D_y \in \mathbb{C}^{m\times m}$ is given by

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

Projector-splitting integrator applied to the equivalent first-order system

In the following, we first study the performance of the projector-splitting integrator when it is applied to the equivalent first-order system (4.2) of (7.7). For $m = 4096$ and $n = 512$ discretization points in y- and x-direction, respectively, we compute a numerical approximation with $r_{\mathbf{A}} = 20$ and different multiples of the step size $\tau_0 = L_y/(160m)$. The results are displayed in Figure 7.4 (left picture). While for $\tau = \tau_0$ the relative global error remains small, the error for the larger step sizes grows rapidly w.r.t. t. Also, the larger the step sizes, the earlier the numerical approximation with the projector-splitting integrator becomes unstable. Hence, even for the homogeneous case, the naive approach for computing a dynamical low-rank approximation based on the first-order formulation (4.2) of (7.7) and the projector-splitting integrator fails. This emphasizes the necessity of designing dynamical low-rank integrators for second-order matrix differential equations differently.

Behavior of `stlo` and `stlostiff` for larger step sizes

A second aspect we examine is the behavior of the `stlo` and `stlostiff` schemes for larger step sizes, without effects caused by nonlinear terms. For the same number of discretization points as above, we compute the relative global error in $\mathbf{A}$ between the exact solution of (7.7) and the low-rank approximations computed with $r_{\mathbf{A}} = r_{\mathbf{B}} = 20$ at $T = 45\pi$. The results of this experiment are shown in Figure 7.4 (right picture). Clearly, the `stlostiff` scheme does not only yield significantly smaller errors, but also allows for larger step sizes compared to the `stlo` scheme. Hence, the application of a dynamical low-rank integrator which exploits the structure of the right-hand side of (7.7) is beneficial in this experiment.

7.2.2 Laser-plasma interaction

As a second example for second-order problems, we consider a reduced model of laser-plasma interaction from [Karle et al., 2006, 2008; Schweitzer, 2008]. It is given by a wave equation with space-dependent cubic nonlinearity on a rectangular domain Ω given in (7.1) with periodic boundary conditions. Discretization in space as is Section 7.2.1 yields the second-order matrix differential equation

$$A''(t) = \mathcal{L}A(t) - 0.3\boldsymbol{\chi} \bullet \big(A(t) - \frac{1}{2}A(t) \bullet \overline{A(t)} \bullet A(t)\big) = F\big(A(t)\big), \qquad A(0) = A_0, \quad A'(0) = B_0. \quad (7.8)$$

As initial values we use

$$(A_0)_{ij} = 0.12 \exp\Big(-\frac{y_i^2}{l_0^2} - \frac{x_j^2}{w_0^2} + \mathrm{i}y_i\Big), \quad (B_0)_{ij} = \Big(-\frac{2y_i}{l_0^2} - \mathrm{i}\Big)(A_0)_{ij}, \qquad i = 1,\dots,m, \quad j = 1,\dots,n.$$

As pointed out in [Karle et al., 2006], this choice of initial conditions turns the solution to the problem (7.8) highly oscillatory in longitudinal direction. Equation (7.8) models the propagation of a laser pulse of 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 $\boldsymbol{\chi} \in \mathbb{R}^{m\times n}$ with entries

$$\boldsymbol{\chi}_{ij} = \begin{cases} 1, & 50\pi \le y_i \le 300\pi, \\ 0, & \text{else}, \end{cases}$$

and is thus of the form as in Section 5.1.3 with $\mu = 1$ and $\nu = n$. The interaction between the pulse and the plasma is modeled by a cubic nonlinearity. As in [Karle et al., 2008] we use the parameters $\lambda_0 = \pi$, $l_0 = 10\pi$, $w_0 = 100\pi$, $L_x = 300\pi$, and $L_y = 600\pi$.

For $m = 8192$ and $n = 1024$ discretization points in longitudinal and transversal direction, respectively, we compute fixed and variable low-rank approximations to the solution of (7.8) for different step sizes. The reference solution was computed with the Gautschi-type method studied in [Schweitzer, 2008] and step size $\tau_0 = L_y/(80m)$. For the low-rank approximations we used $\tau = 2^k\tau_0$, $k = 2,\dots,6$. All fixed-rank integrators were performed with $r_{\mathbf{A}} = r_{\mathbf{B}} = 4$. For the methods `stlostiff` and `rastlostiff` we used the weights

$$\omega_1^2 = \frac{2}{3}, \quad \omega_2^2 = 0, \quad \omega_3^2 = \frac{1}{3},$$

which is motivated by the fact that the pulse mainly travels along the y-axis, i.e., in longitudinal direction.

The top row in Figure 7.5 shows the relative global errors in $\mathbf{A}$ and $\mathbf{B}$ at $T = 600\pi$ between the reference solution and the low-rank approximations. For comparison, we also computed approximations with the leapfrog scheme. As one can observe, all integrators show convergence of order 2. For the convergence in $\mathbf{A}$, the curves for the `stlo`, `rastlo`, `stlotangent`, `rastlotangent`, `stlotwostep`, and `leapfrog` schemes align almost perfectly, whereas the `leapfrog` method yields slightly smaller errors in $\mathbf{B}$ for small step sizes. The accuracy of the low-rank methods and the leapfrog scheme is comparable, but the dynamical low-rank integrator provide the results with significantly less storage requirements. Instead of storing arrays of size $m \times n$, we only need to store matrices of dimension $m \times r_{\mathbf{A}}$, $n \times r_{\mathbf{A}}$, and

$r_{\mathbf{A}} \times r_{\mathbf{A}}$. For the chosen values for m, n, and $r_{\mathbf{A}}$ this means that the low-rank schemes require less than 0.5% of the memory the leapfrog scheme needs. The rank-adaptive dynamical low-rank integrators choose approximation ranks up to $r = 3$ (`stlo` and `stlotangent` schemes) and $r = 4$ (`stlostiff` scheme).

The `gautschilr` scheme yields smaller errors than the methods listed above, and the `stlostiff` and `rastlostiff` methods, whose curves are almost indistinguishable, yield even smaller errors. This behavior is explained by the construction of these methods, where the structure of the right-hand side is better taken into account than in the other schemes, cf. Section 4.3 and Section 4.4.2.

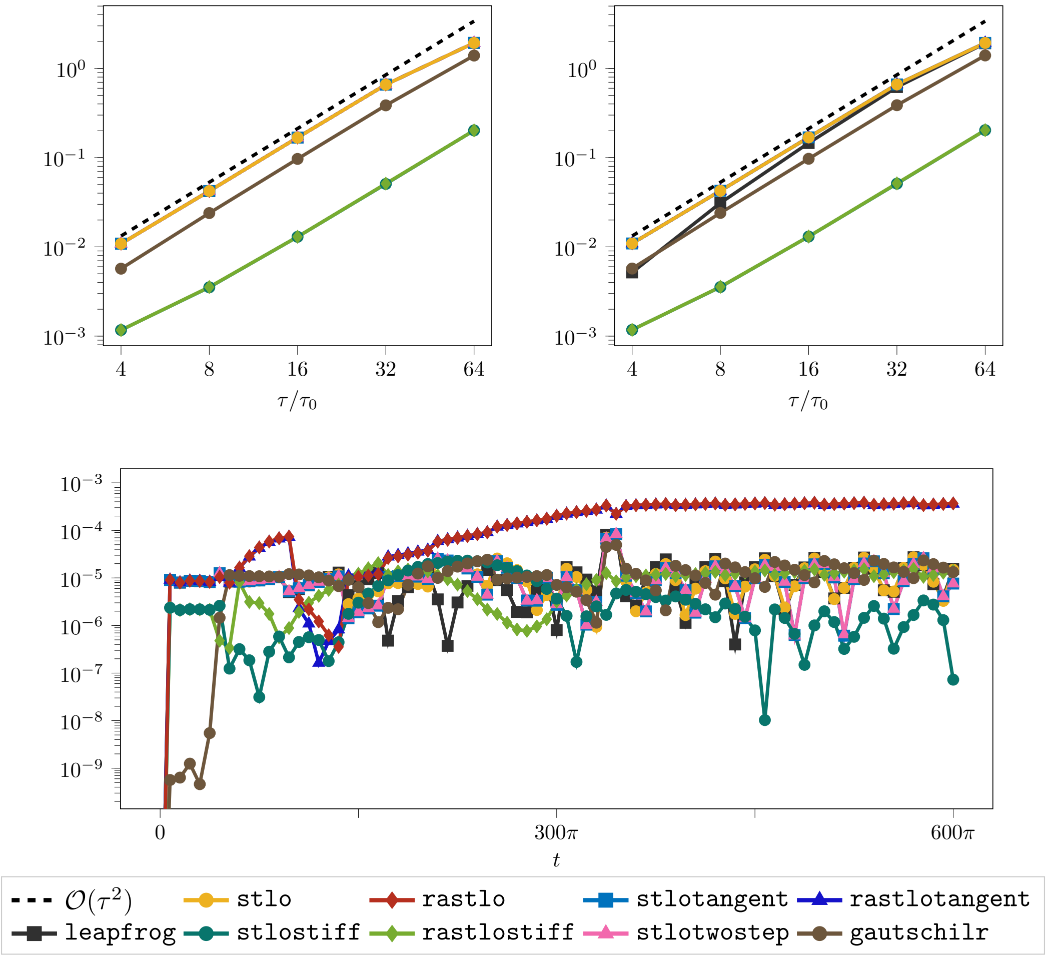

Figure 7.5. Laser-plasma interaction. **Top:** Relative global error in $\mathbf{A}$ (left) and $\mathbf{B}$ (right) between reference solution and low-rank approximations at $T = 600\pi$. **Bottom:** Relative error in the maximal intensity for $\tau = 4\tau_0$. The fixed-rank methods were computed with $r_{\mathbf{A}} = r_{\mathbf{B}} = 4$, the rank-adaptive methods with $M = 100$.

In physics, the maximal intensity

$$\max_{i,j} |A_{ij}(t)|^2$$

of the propagating pulse over time is sometimes of higher interest that A itself. The bottom picture of Figure 7.5 shows the relative error between the maximal intensity of the reference solution and the maximal intensity of the low-rank methods and the leapfrog scheme.

Figure 7.6 shows the absolute value and the real part of the reference solution and the low-rank approximations computed with the `stlo` and `stlostiff` schemes with step size $\tau = 4\tau_0$ at $t = 37.5\pi$

and $x = 0$. The left picture shows the laser pulse, which moves to the right, while the right picture shows a small reflection moving into the opposite direction. In both pictures, the high oscillations are clearly visible. The curves for the three methods in the left picture are nearly indistinguishable. The right picture shows very small differences in the approximation of the reflection (note the scale of 10^{-7} which is below the tolerance threshold): Neither the `stlo` nor the `stlostiff` scheme compute a reflection of the same absolute value as the Gautschi reference solution. However, the phase of the oscillations is captured well by the `stlostiff` scheme, while the `stlo` method shows some phase drift.

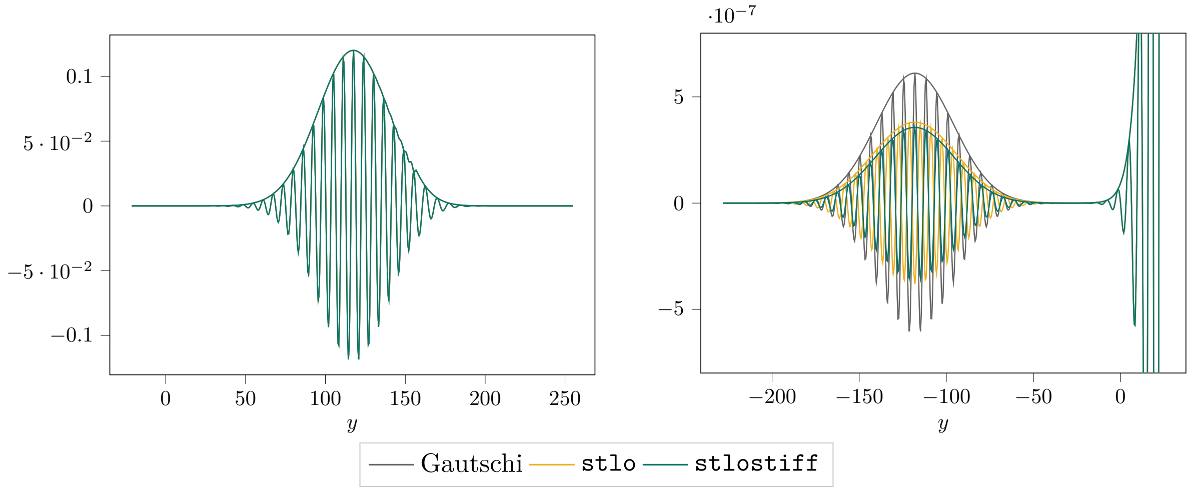

Figure 7.6. Laser-plasma interaction. Absolute value and real part of reference solution and low-rank approximations computed with the `stlo` and `stlostiff` schemes at $t = 37.5\pi$, $x = 0$, and step size $\tau = 4\tau_0$.

The fact that the accuracy of the rank-adaptive integrators is comparable to the one of their fixed-rank variants indicates, that the heuristics for determining the tolerance threshold also works nicely for this experiment. The trajectories of the singular values of the rank-adaptive integrators together with the first ten singular values of the reference solution and the respective tolerance thresholds are presented in Figure 7.7.

7.2.3 Sine-Gordon equation

In our last experiment we consider the two-dimensional sine-Gordon equation on the domain Ω from (7.1) with homogeneous Neumann boundary conditions, cf. [Bratsos, 2005]. Using finite differences of order two on the grid (7.3) with $L_x = L_y = 7$ and $m = n = 1001$ discretization points in both x- and y-direction, we obtain the semi-discretized second-order matrix differential equation

$$A''(t) = DA(t) + A(t)D^T - \Phi \bullet \underline{\sin}\big(A(t)\big), \quad t \in [0,T], \qquad A(0) = A_0, \quad A'(0) = B_0.$$

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

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

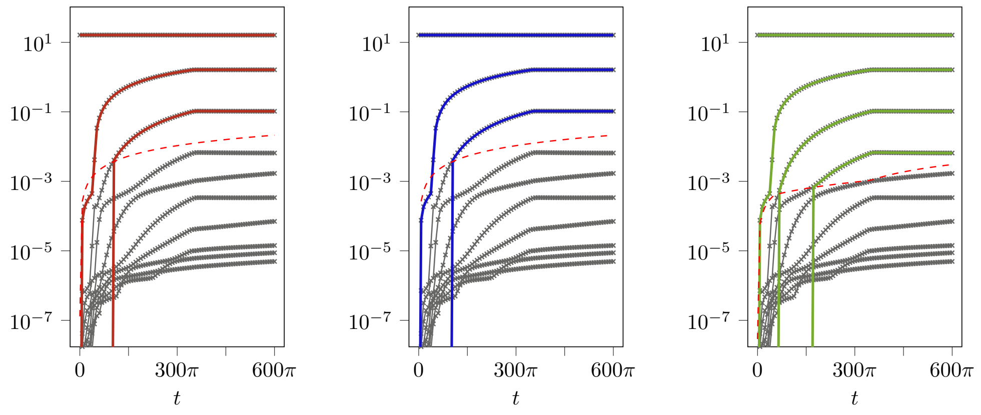

Figure 7.7. Laser-plasma interaction. Trajectories of the ten largest singular values of the reference solution (gray) together with the trajectories of the singular values of the rank-adaptive dynamical low-rank integrators (left: `rastlo`, middle: `rastlotangent`, right: `rastlostiff`) for $\tau = 4\tau_0$ and the respective computed tolerance thresholds (red, dashed).

According to [Strang, 1999], D is diagonalizable by the discrete cosine transform (DCT-1). This allows an efficient implementation by diagonalization of the matrix functions in the `stlostiff` and `rastlostiff` methods. For these we use the weights

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

since there is no preferred direction of propagation.

For our first experiment, we use the initial values

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

and

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

$i, j = 0, \dots, m$. This particular choice yields a line soliton in an inhomogeneous medium [Bratsos, 2007, Section 3.1.3]. The reference solution is computed by the leapfrog scheme on the same spatial grid with step size $\tau_0 = 2.5 \cdot 10^{-5}$.

The top row of Figure 7.8 shows the relative global errors in $\mathbf{A}$ and $\mathbf{B}$ between the low-rank approximations and the reference solution. Convergence order two is observed for all methods. For the approximations in $\mathbf{A}$, the fixed-rank integrators are slightly more accurate than their rank-adaptive pendants, probably because they use a higher rank. For the approximations in $\mathbf{B}$, the approximation quality of the rank-adaptive versions is as good as for the fixed-rank integrators. For comparison, we also computed approximations with the leapfrog scheme. As in the laser-plasma interaction the errors induced by the leapfrog scheme are not significantly smaller than the ones by the low-rank methods.

The bottom row of Figure 7.8 displays the evolution of the approximation rank $r_{\mathbf{A}}$ for $\tau = 160\tau_0$ and $\tau = 4\tau_0$, respectively. As one observes, the rank is small at the beginning but grows with ongoing time. Also, for $\tau = 160\tau_0$ approximations are computed with at most 10 singular values, and hence the storage effort is roughly halved compared to the fixed-rank schemes.

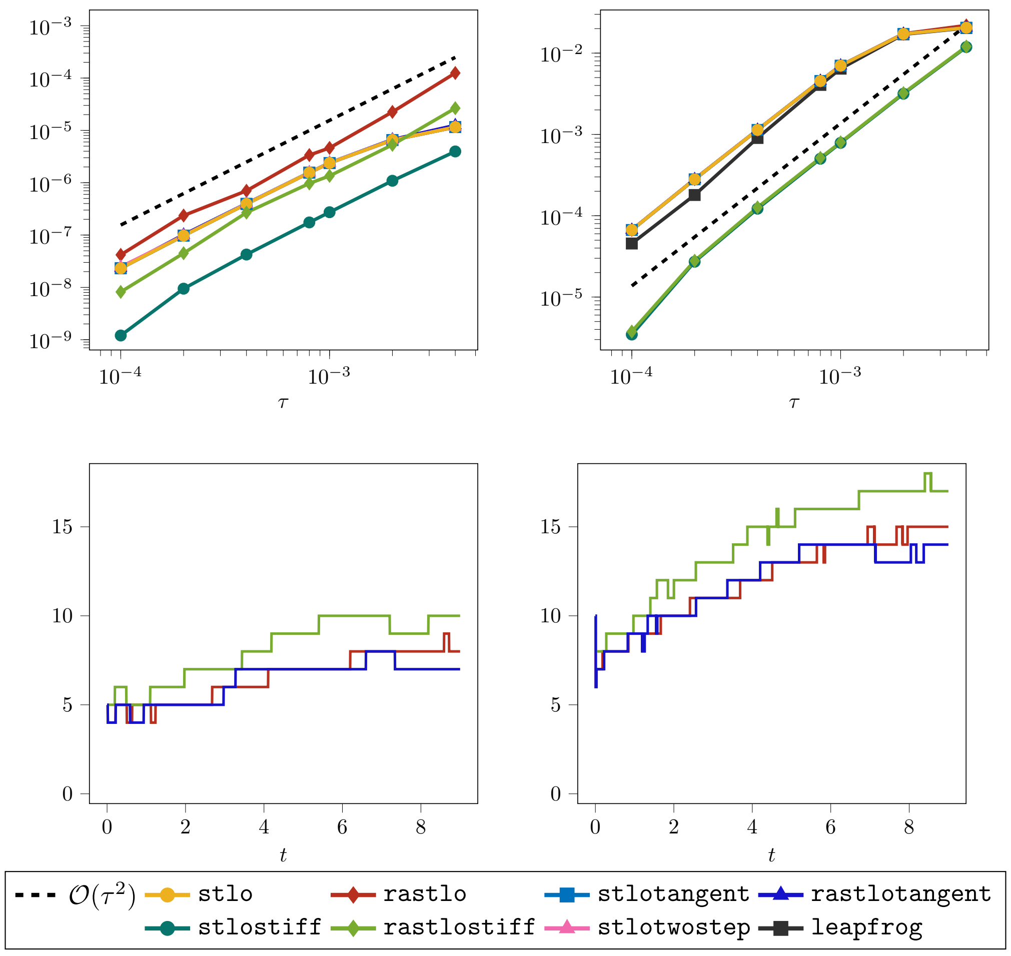

Figure 7.8. Sine-Gordon equation, first setting. **Top:** Relative global error in $\mathbf{A}$ (left) and $\mathbf{B}$ (right) between reference solution and low-rank approximations at $T = 9$. **Bottom:** Rank evolution of the approximations $\mathbf{A}$ computed with the rank-adaptive dynamical low-rank integrators for $\tau = 160\tau_0$ (left) and $\tau = 4\tau_0$ (right). For the fixed-rank integrators, we used $r_\mathbf{A} = r_\mathbf{B} = 20$, for the rank-adaptive methods $M = 100$.

In a second setting, we consider the symmetric perturbation of a static line soliton [Bratsos, 2007, Section 3.1.2] with $\Phi_{ij} = 1$, $(B_0)_{ij} = 0$, and

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

Figure 7.9 shows a similar behavior of the methods as in the first setting. For the rank-adaptive schemes and $\tau = 4\tau_0$ however, the initial rank is rather large, and drops significantly after a few steps. As explained in Remark 6.2, 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 particular experiment, an initial rank of ~ 23 is sufficient. The guesses 5, 10, and 20 are hence rejected, until $r_0 = 40$ is accepted and rank-reduction applies in the subsequent integration steps.

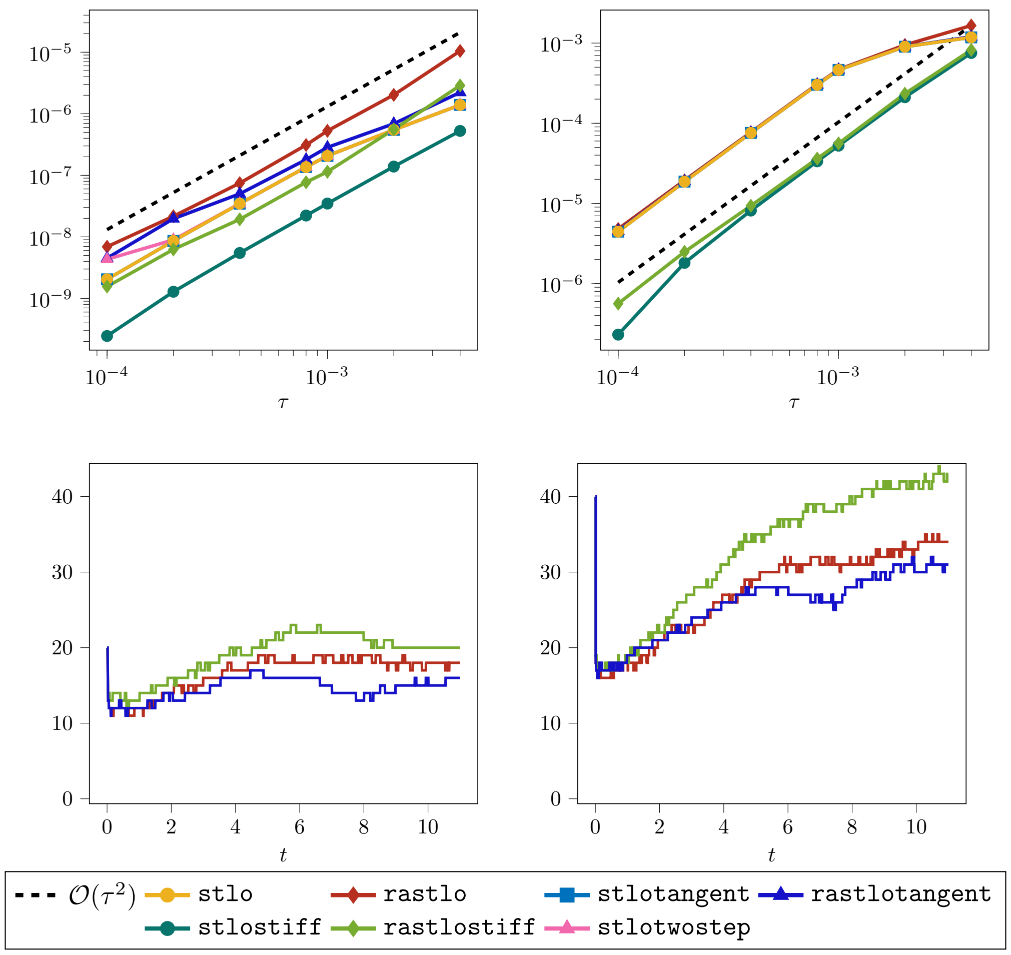

Figure 7.9. Sine-Gordon equation, second setting. **Top:** Relative global error in $\mathbf{A}$ (left) and $\mathbf{B}$ (right) between reference solution and low-rank approximations at $T = 11$. **Bottom:** Rank evolution of the approximations $\mathbf{A}$ computed with the rank-adaptive dynamical low-rank integrators for $\tau = 160\tau_0$ (left) and $\tau = 4\tau_0$ (right). For the fixed-rank integrators, we used $r_{\mathbf{A}} = r_{\mathbf{B}} = 50$, for the rank-adaptive methods $M = 100$.

APPENDIX A

Collection of results from linear algebra

In this chapter, we briefly collect some definitions and results of linear algebra we use throughout this thesis. If not stated otherwise, the following definitions and theorems are taken from [Trefethen and Bau, 1997] and [Horn and Johnson, 1990].

A.1 The singular value decomposition

We start this section with the definition of the *rank* of a matrix:

Definition A.1. *The* **column rank** *of a matrix is the dimension of its column space, e.g., the space spanned by its columns. Similarly, the* **row rank** *of a matrix is the dimension of its row space.*

For each matrix, column rank and row rank are equal, so that we mostly do not distinguish between column and row rank. We call a $m \times n$ a matrix of **full rank**, of is has the maximal possible rank $\min\{m, n\}$. For the rank of the sum and the product of matrices, the following theorem holds:

Theorem A.2. *Let $A, B \in \mathbb{C}^{m\times n}$ and $C \in \mathbb{C}^{n\times k}$ with arbitrary $m, n, k \in \mathbb{N}$. Then*

1. $\operatorname{rank}(A + B) \leq \operatorname{rank}(A) + \operatorname{rank}(B)$
2. $\operatorname{rank}(AC) \leq \min\{\operatorname{rank}(A), \operatorname{rank}(C)\}$

We now introduce the singular value decomposition, which is often referred to as “full” singular value decomposition in the literature. In the following, we call a matrix $U \in \mathbb{C}^{m\times n}$ unitary if its columns are orthonormal.

Definition A.3. *Let m and n be arbitrary natural numbers. Given $A \in \mathbb{C}^{m\times n}$, not necessarily of full rank, a* **singular value decomposition** *of A is a factorization*

$$A = \widehat{U}\widehat{\Sigma}\widehat{V}^H \tag{A.1}$$

where

$$\begin{aligned} \widehat{U} &\in \mathbb{C}^{m\times m} && \textit{is unitary,}\\ \widehat{V} &\in \mathbb{C}^{n\times n} && \textit{is unitary,}\\ \widehat{\Sigma} &\in \mathbb{R}^{m\times n} && \textit{is diagonal.} \end{aligned}$$

In addition, it is assumed that the diagonal entries σ_j of $\widehat{\Sigma}$ are nonnegative and in nonincreasing order, e.g., $\sigma_1 \geq \ldots \geq \sigma_k \geq 0$ where $k = \min\{m, n\}$.

For the singular value decomposition, the following uniqueness property holds:

Theorem A.4. *Every matrix $A \in \mathbb{C}^{m\times n}$ has a singular value decomposition (A.1). Furthermore, the singular values $\{\sigma_j\}$ are uniquely determined, and, if A is square and the σ_j are distinct, the left and right singular vectors $\{\widehat{U}_j\}$ and $\{\widehat{V}_j\}$ are uniquely determined up to complex signs (i.e., complex scalar factors of absolute value 1).*

Besides the full singular value decomposition, there is a reduced variant:

Definition A.5. *Let $m \geq n$ and consider $A \in \mathbb{C}^{m\times n}$ with $\operatorname{rank} A = p \leq n$. Then the* **reduced singular value decomposition** *of A reads*

$$A = U\Sigma V^H, \tag{A.2}$$

where $U \in \mathbb{C}^{m\times p}$ and $V \in \mathbb{C}^{n\times p}$ are unitary matrices, and $\Sigma \in \mathbb{R}^{p\times p}$ is a square diagonal matrix with positive real entries.

Next, we introduce the Frobenius and spectral norm for matrices:

Definition A.6. *The* **Frobenius norm** *of a matrix $A \in \mathbb{C}^{m\times n}$ is defined as*

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |A_{ij}|^2},$$

and the **spectral norm** *norm of A is defined as*

$$\|A\|_2 = \max_{\|x\|_2} \|Ax\|_2, \quad x \in \mathbb{C}^n.$$

In the following theorems we collect some properties of the Frobenius and the spectral norms.

Theorem A.7. *Let $A \in \mathbb{C}^{m\times n}$ and let $p \leq \min\{m, n\}$ denote the number of nonzero singular values σ_i of A. Furthermore, denote by $\operatorname{span}\{x, y, \ldots, z\}$ the space spanned by the vectors $x, y, \ldots, z$. Then the following holds:*

1. *The rank of A is p.*
2. $\|A\|_F^2 = \operatorname{tr}(A^H A) = \operatorname{tr}(AA^H)$.
3. $\|A\|_F^2 = \sigma_1^2 + \ldots + \sigma_p^2$.
4. $\|A\|_2 = \sigma_1$.

5. $\operatorname{range}(A) = \operatorname{span}\{U_1, \ldots, U_p\}$, *where U_j denotes the jth column of U in* (A.2).

6. *A is the sum of r rank-one matrices:*
$$A = \sum_{j=1}^{p} \sigma_j U_j V_j^H.$$

7. *For any unitary $U \in \mathbb{C}^{\ell \times m}$, $\ell \geq m$, and any unitary $V \in \mathbb{C}^{n \times k}$, $n \geq k$, we have*
$$\|UAV^H\|_2 = \|A\|_2, \qquad \|UAV^H\|_F = \|A\|_F.$$

8. *For any $B \in \mathbb{C}^{m \times n}$,*
$$\|A - B\|_F^2 = \|A\|_F^2 - 2 \operatorname{Re} \operatorname{tr}(A^H B) + \|B\|_F^2.$$

Given a matrix $A \in \mathbb{C}^{m \times n}$ and an approximation rank r, the singular value decomposition allows to determine the rank-r best-approximation to A:

Theorem A.8. *For any r with $0 \leq r \leq p = \operatorname{rank} A$, define*
$$A_r = \sum_{j=1}^{r} \sigma_j U_j V_j^H.$$
Then
$$\|A - A_r\|_F = \inf_{\substack{B \in \mathbb{C}^{m \times n} \\ \operatorname{rank}(B) \leq r}} \|A - B\|_F = \sqrt{\sigma_{r+1}^2 + \ldots + \sigma_p^2}.$$
A_r is also called the **rank-r best-approximation** *to A.*

A.2 The QR factorization

Definition A.9 ([Trefethen and Bau, 1997, Lecture 7])**.** *Let $A \in \mathbb{C}^{m \times n}$. Then there exists a factorization*
$$A = QR,$$
where $Q \in \mathcal{V}_{m,n}$ and R is an $n \times n$ upper triangular matrix. Such a factorization is called a **reduced QR factorization/decomposition of A***.*

The QR factorization is not unique:
$$A = Q_1 R_1 = (Q_1 D)(D^H R_1) = Q_2 R_2,$$
where $D \in \mathbb{C}^{n \times n}$ is diagonal and satisfies $D^H D = DD^H = I$.

A.3 Kronecker products and sums

This section is devoted to the Kronecker products and sums, and related topics.

Definition A.10. *Let $A \in \mathbb{C}^{m\times n}$ and $B \in \mathbb{C}^{p\times q}$. The* **Kronecker product** *$A \otimes B$ is the $mp \times nq$ block matrix*

$$A \otimes B = \begin{bmatrix} A_{11}B & A_{12}B & \cdots & A_{1n}B \\ A_{21}B & A_{22}B & \cdots & A_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1}B & A_{m2}B & \cdots & A_{mn}B \end{bmatrix}.$$

If $A \in \mathbb{C}^{m\times m}$ and $B \in \mathbb{C}^{n\times n}$, the **Kronecker sum** *is defined as*

$$A \oplus B = I_n \otimes A + B \otimes I_m.$$

Theorem A.11 (Properties of the Kronecker product, cf. [Van Loan, 2000, Section 1])**.**

1. *Let $A, B, C \in \mathbb{C}^{m\times n}$ and $c \in \mathbb{C}$ a scalar. Then the following identities hold:*

$$\begin{aligned} A \otimes (B + C) &= A \otimes B + A \otimes C, \\ (A + B) \otimes C &= A \otimes C + B \otimes C, \\ (cA) \otimes B &= A \otimes (cB) = c(A \otimes B), \\ (A \otimes B) \otimes C &= A \otimes (B \otimes C), \\ (A \otimes B)^T &= A^T \otimes B^T, \\ (A \otimes B)^H &= A^H \otimes B^H. \end{aligned}$$

   If A and B are square and invertible, we also have

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}.$$

2. *Let A, B, C and D be matrices of such sizes that the products AC and BD can be formed. Then*

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD).$$

Definition A.12. *Let $A \in \mathbb{C}^{m\times n}$. The map $\operatorname{vec} : \mathbb{C}^{m\times n} \to \mathbb{C}^{mn}$ which maps A to a vector by stacking the columns of A above each other,*

$$\operatorname{vec} A = \begin{bmatrix} A_{11} & \cdots & A_{m1} & A_{12} & \cdots & A_{m2} & \cdots & A_{1n} & \cdots & A_{mn} \end{bmatrix}^T,$$

is called **vectorization** *of A. Its inverse map $\operatorname{vec}_{m,n}^{-1} : \mathbb{C}^{mn} \to \mathbb{C}^{m\times n}$ is called* **devectorization**. *Note that for the devectorization the dimension of the output matrix needs to be stated explicitly.*

Theorem A.13. *Let $A, B \in \mathbb{C}^{m\times n}$ and $\alpha \in \mathbb{C}$. The vectorization map vec satisfies the following properties:*

1. $\operatorname{vec}(A + B) = \operatorname{vec} A + \operatorname{vec} B$,
2. $\operatorname{vec}(\alpha A) = \alpha \operatorname{vec} A$,
3. $\|A\| = \|\operatorname{vec} A\|_2$.

The Kronecker product and vectorization are compatible:

Theorem A.14 ([Henderson and Searle, 1980/81, Section 2.3])**.** *For arbitrary matrices $A \in \mathbb{C}^{m\times n}$, $B \in \mathbb{C}^{n\times k}$, $C \in \mathbb{C}^{k\times p}$, it holds*

$$\operatorname{vec}(ABC) = (C^T \otimes A) \operatorname{vec} B.$$

The Kronecker sum plays an important role in the solution of particular matrix equations, e.g., the Sylvester equation:

Definition A.15 ([Van Loan, 2000, Section 2])**.** *Let $L_1 \in \mathbb{C}^{m\times m}$, $L_2 \in \mathbb{C}^{n\times n}$, and $A, X \in \mathbb{C}^{m\times n}$. The* **Sylvester equation** *is a matrix equation of the form*

$$X = L_1 A + A L_2.$$

With the vectorization operator vec*, this equation can be written as*

$$\operatorname{vec} X = (L_1 \oplus L_2^T) \operatorname{vec} A = (I_n \otimes L_1 + L_2^T \otimes I_m) \operatorname{vec} A.$$

We end this section with a theorem concerning the eigenvalues of a Kronecker sum.

Theorem A.16 ([Horn and Johnson, 1994, Section 4.4])**.** *Let $A \in \mathbb{C}^{m\times m}$ and $B \in \mathbb{C}^{n\times n}$ have the eigenvalues λ_i, $i = 1, \dots, m$ and μ_j, $j = 1, \dots, n$, respectively. Then the Kronecker sum $A \oplus B$ has the eigenvalues $\lambda_i + \mu_j$, $i = 1, \dots, m$, $j = 1, \dots, n$.*

A.4 Special matrices

Next, we consider special matrices which appear in this thesis.

Definition A.17. *A* **Toeplitz matrix** *is a matrix $A \in \mathbb{C}^{m\times n}$, in which each diagonal from left to right is constant. If $m = n$, then A has the form*

$$A = \begin{pmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-m+1} \\ a_1 & a_0 & a_{-1} & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & & \ddots & a_1 & a_0 & a_{-1} \\ a_{m-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{pmatrix}.$$

A **circulant matrix** *$C(z) \in \mathbb{C}^{m\times m}$ is a square Toeplitz matrix which is fully specified by the first column z. The remaining columns of $C(z)$ are cyclic permutations of z:*

$$C(z) = \begin{pmatrix} z_0 & z_{m-1} & \cdots & z_2 & z_1 \\ z_1 & z_0 & z_{m-1} & & z_2 \\ \vdots & z_1 & z_0 & \ddots & \vdots \\ z_{m-2} & & \ddots & \ddots & z_{m-1} \\ z_{m-1} & z_{m-2} & \cdots & z_1 & z_0 \end{pmatrix}.$$

Theorem A.18 ([Golub and Van Loan, 2013, Theorem 4.8.2])**.** *A circulant matrix $C(z) \in \mathbb{C}^{m\times m}$ is Fourier-diagonalizable, i.e.,*

$$C(z) = \mathcal{F}_m^{-1}\Lambda\mathcal{F}_m, \qquad \Lambda = \operatorname{diag}(\mathcal{F}_m z).$$

Here, $\mathcal{F}_m$ denotes the discrete Fourier transform matrix.

Definition A.19. *An* **upper Hessenberg matrix** *is a matrix $H \in \mathbb{C}^{m\times m}$ whose entries are zero below the first subdiagonal, i.e.,*

$$H = \begin{pmatrix} h_{1,1} & h_{1,2} & h_{1,3} & \cdots & h_{1,m} \\ h_{2,1} & h_{2,2} & h_{2,3} & \cdots & h_{2,m} \\ 0 & h_{3,2} & h_{3,3} & \cdots & h_{3,m} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{m,m-1} & h_{m,m} \end{pmatrix}.$$

We call H **unreduced***, if the matrix has no zero subdiagonal entries.*

A.5 Matrix functions

We end this chapter with some notes on matrix functions.

Definition A.20 (Matrix functions)**.** *Let the power series*

$$g(z) = \sum_{k=0}^{\infty} a_k z^k$$

possess the radius of convergence $\varrho > 0$. Assume

$$\rho(L) = \max\{|\lambda| \mid \lambda \textit{ eigenvalue of } L\} < \varrho,$$

where $\rho(L)$ denotes the spectral radius of the matrix $L \in \mathbb{C}^{m\times m}$. Then we define the **matrix function**

$$g(L) := \sum_{k=0}^{\infty} a_k L^k.$$

In the next theorem, we collect some properties of matrix functions.

Theorem A.21 ([Higham, 2008, Theorem 1.13])**.** *Let $L \in \mathbb{C}^{m\times m}$ and let*

$$g(z) = \sum_{k=0}^{\infty} a_k z^k$$

be the power series of g with radius of convergence $\varrho > \rho(L)$. Then the following holds:

1. *The matrix function $g(L)$ commutes with L, i.e., $g(L)L = Lg(L)$.*

2. *$g(L)^T = g(L^T)$.*

3. *If $a_k \in \mathbb{R}$ for all $k \in \mathbb{N}_0$, then $g(L)^H = g(L^H)$.*

4. *If L is an upper (or lower) block triangular matrix, so is $G = g(L)$. In addition, $G_{ii} = g(L_{ii})$.*

5. *Let*

$$L = S_1 \widetilde{L} S_2, \qquad \widetilde{L} \in \mathbb{C}^{\ell\times\ell}, \quad S_1 \in \mathbb{C}^{m\times\ell}, \quad S_2 \in \mathbb{C}^{\ell\times m},$$

where $S_2 S_1 = I_\ell$. Then $g(L) = S_1 g(\widetilde{L}) S_2$.

Theorem A.22 ([Higham, 2008, Section 2.1])**.** *Let $\Omega \in \mathbb{C}^{m\times m}$ be a Hermitian, positive semidefinite matrix. Then it holds*

$$\exp\left(t\begin{bmatrix} 0 & I \\ -\Omega^2 & 0\end{bmatrix}\right) = \begin{bmatrix} \cos(t\Omega) & t\,\mathrm{sinc}(t\Omega) \\ -t\Omega^2\,\mathrm{sinc}(t\Omega) & \cos(t\Omega)\end{bmatrix}.$$

Corollary A.23. *Let $\Omega \in \mathbb{C}^{m\times m}$ be a Hermitian, positive semidefinite matrix and $\omega > 0$ a scalar. Then*

$$\exp\left(t\begin{bmatrix} 0 & \omega^2 I \\ -\Omega^2 & 0\end{bmatrix}\right) = \begin{bmatrix} \cos(\omega t\Omega) & \omega^2 t\,\mathrm{sinc}(\omega t\Omega) \\ -t\Omega^2\,\mathrm{sinc}(\omega t\Omega) & \cos(\omega t\Omega)\end{bmatrix}.$$

Proof. We rewrite the argument of the matrix exponential as

$$\exp\left(t\begin{bmatrix} 0 & \omega^2 I \\ -\Omega^2 & 0\end{bmatrix}\right) = \exp\left((\omega^2 t)\begin{bmatrix} 0 & I \\ -\left(\frac{1}{\omega}\Omega\right)^2 & 0\end{bmatrix}\right).$$

The assertion now directly follows from Theorem A.22. □

APPENDIX B

A short note on splitting methods

A fundamental concept which we use throughout this thesis in order to solve first-order or second-order matrix differential equations are *splitting methods*. They are essential in the construction of all dynamical low-rank integrators. Here, we give a brief overview of the two most relevant splitting schemes and their derivation. More details on (general) splitting schemes can be found in [Hairer et al., 2006, Section II.5].

We first introduce the notion of a *flow* over time:

Definition B.1 ([Hairer et al., 2006, Section I.1.1])**.** *Consider the first-order differential equation*

$$a'(t) = f\big(a(t)\big), \quad t \in [0, T], \qquad a(0) = a_0, \tag{B.1}$$

where $a(t)$ might be a scalar, a vector, or a matrix. The **flow** *of this differential equation is the mapping, which associates the value $a(t)$ of the solution with initial value $a(0) = a_0$. This map, denoted by φ_t is thus defined by*

$$\varphi_t(a_0) = a(t) \quad \textit{if} \quad a(0) = a_0.$$

Assume now that the right-hand side of the first-order differential equation (B.1) can be split into

$$a'(t) = f_1\big(a(t)\big) + f_2\big(a(t)\big),$$

and the exact flows $\varphi_t^{[f_1]}$ and $\varphi_t^{[f_2]}$ of the equations $a' = f_1(a)$ and $a' = f_2(a)$, respectively, can be calculated explicitly. Then, for a given step size $\tau > 0$, starting from the initial value a_0, one can solve the first subproblem in $[0, \tau]$ to obtain a value $a_{\frac{1}{2}}$, and from this value solve the second subproblem in $[0, \tau]$ to obtain an approximation $a_1 \approx a(\tau)$. The flow ϕ_t of the resulting numerical method reads

$$\phi_\tau = \varphi_\tau^{[f_2]} \circ \varphi_\tau^{[f_1]},$$

and is often called the *Lie-Trotter splitting*. This method gives approximations of order 1 to the exact solution of (B.1), which is easily seen by Taylor series expansion. Alternatively, one could exchange the flows, which yields

$$\phi_\tau^* = \varphi_\tau^{[f_1]} \circ \varphi_\tau^{[f_2]},$$

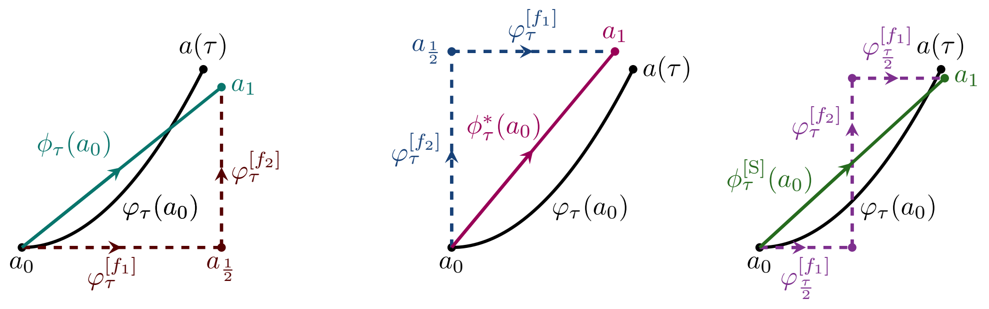

Figure B.1. Graphical illustration of the Lie-Trotter splitting (left), its adjoint (middle), and the Strang splitting (right).

which is the *adjoint* of the first method:

Definition B.2 ([Hairer et al., 2006, Section II.3, Definition 3.1])**.** *The* **adjoint method** ϕ_τ^* *of a method* ϕ_τ *is the inverse of the original method with reversed time step* $-\tau$*, i.e.,*

$$\phi_\tau^* := \phi_{-\tau}^{-1}.$$

In other words, $a_1 = \phi_\tau^*(a_0)$ *is implicitly defined by* $\phi_{-\tau}(a_1) = a_0$*. A method for which* $\phi_\tau^* = \phi_\tau$ *is called* **symmetric***.*

A symmetric version is defined via

$$\phi_\tau^{[S]} = \varphi_{\frac{\tau}{2}}^{[f_1]} \circ \varphi_\tau^{[f_2]} \circ \varphi_{\frac{\tau}{2}}^{[f_1]},$$

which is known as *Strang splitting*. It can also be seen as a composition of the Lie-Trotter splitting with its adjoint with halved step size, since

$$\phi_{\frac{\tau}{2}}^* \circ \phi_{\frac{\tau}{2}} = \varphi_{\frac{\tau}{2}}^{[f_1]} \circ \varphi_{\frac{\tau}{2}}^{[f_2]} \circ \varphi_{\frac{\tau}{2}}^{[f_2]} \circ \varphi_{\frac{\tau}{2}}^{[f_1]} = \phi_\tau^{[S]}.$$

Due to its symmetry, the Strang splitting yields approximations of order 2.

Both approaches for constructing the Lie-Trotter splitting and the Strang splitting, respectively, can be generalized: Firstly, if the exact flow of say the second subproblem is not known, it can be replaced by the numerical flow $\phi_t^{[f_2]}$ of an arbitrary numerical method applied to $a' = f_2(a)$. The Lie-Trotter splitting then reads

$$\phi_\tau = \phi_\tau^{[f_2]} \circ \varphi_\tau^{[f_1]}. \tag{B.2}$$

If the flow $\phi_t^{[f_2]}$ yields approximations of at least order 1, (B.2) still is a first-order method.

If the right-hand side of (B.1) is given as

$$a' = f_1(a) + f_2(a) + \ldots + f_N(a),$$

for some $N \in \mathbb{N}$, the Lie-Trotter splitting takes the form

$$\phi_\tau = \varphi_\tau^{[f_N]} \circ \ldots \circ \varphi_\tau^{[f_2]} \circ \varphi_\tau^{[f_1]},$$

while the Strang splitting then reads

$$\phi_{\tau}^{[\mathrm{S}]} = \varphi_{\frac{\tau}{2}}^{[f_1]} \circ \ldots \circ \varphi_{\frac{\tau}{2}}^{[f_{N-1}]} \circ \varphi_{\tau}^{[f_N]} \circ \varphi_{\frac{\tau}{2}}^{[f_{N-1}]} \ldots \circ \varphi_{\frac{\tau}{2}}^{[f_1]}.$$

In order to perform a splitting scheme for solving a second-order differential equation of form

$$a''(t) = f\big(a(t)\big) = f_1\big(a(t)\big) + \ldots + f_N\big(a(t)\big), \quad t \in [0,T], \qquad a(0) = a_0, \quad a'(0) = b_0, \tag{B.3}$$

approximately, we rewrite the differential equation into an equivalent first-order system,

$$\begin{bmatrix} a(t) \\ b(t) \end{bmatrix}' = \begin{bmatrix} b(t) \\ f\big(a(t)\big) \end{bmatrix} = \begin{bmatrix} b(t) \\ f_1\big(a(t)\big) + \ldots + f_N\big(a(t)\big) \end{bmatrix}.$$

The right-hand side can now be rewritten into

$$\begin{bmatrix} b(t) \\ f_1\big(a(t)\big) + \ldots + f_N\big(a(t)\big) \end{bmatrix} = \begin{bmatrix} \alpha_1 b(t) \\ f_1\big(a(t)\big) \end{bmatrix} + \ldots + \begin{bmatrix} \alpha_N b(t) \\ f_N\big(a(t)\big) \end{bmatrix},$$

where we introduced scalar *weights* α_i, $i = 1, \ldots, N$, which satisfy

$$\alpha_1 + \ldots + \alpha_N = 1.$$

Then, the Lie-Trotter splitting and the Strang splitting are defined as above. Formally, the weights are allowed to be any real or complex number. A natural choice for the α_i however is

$$\alpha_i = \frac{1}{N}, \quad i = 1, \ldots, N.$$

Another way of performing a splitting for the second-order differential equation (B.3) is to rewrite the right-hand side of the first-order formulation as

$$\begin{bmatrix} b(t) \\ f\big(a(t)\big) \end{bmatrix} = \begin{bmatrix} 0 \\ f\big(a(t)\big) \end{bmatrix} + \begin{bmatrix} b(t) \\ 0 \end{bmatrix}. \tag{B.4}$$

It can be viewed as a special case of the above, where

$$f_1(a) = f(a), \quad f_2(a) = 0, \qquad \alpha_1 = 0, \quad \alpha_2 = 1.$$

The leapfrog scheme can be derived by performing a Strang splitting based on (B.4), cf. Section 4.1.

Bibliography

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008. URL `https://press.princeton.edu/absil`.

A. H. Al-Mohy and N. J. Higham. Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM J. Sci. Comput.*, 33(2):488–511, 2011. doi:10.1137/100788860.

I. S. Aranson and L. Kramer. The world of the complex ginzburg-landau equation. *Rev. Mod. Phys.*, 74:99–143, 2002. doi:10.1103/RevModPhys.74.99.

N. Boumal. An introduction to optimization on smooth manifolds. Available online, 2020. URL `http://www.nicolasboumal.net/book`.

A. Bratsos. The solution of the two-dimensional sine-gordon equation using the method of lines. *Journal of Computational and Applied Mathematics*, 206(1):251–277, 2007. doi:10.1016/j.cam.2006.07.002.

A. G. Bratsos. An explicit numerical scheme for the sine-Gordon equation in $2+1$ dimensions. *Appl. Numer. Anal. Comput. Math.*, 2(2):189–211, 2005. doi:10.1002/anac.200410035.

C. Carle. *On leapfrog-Chebyshev schemes for second-order differential equations*. PhD thesis, Karlsruhe Institute of Technology, 2021. doi:10.5445/IR/1000147725.

C. Carle, M. Hochbruck, and A. Sturm. On leapfrog-Chebyshev schemes. *SIAM J. Numer. Anal.*, 58(4):2404–2433, 2020. doi:10.1137/18M1209453.

C. Çelik and M. Duman. Crank-Nicolson method for the fractional diffusion equation with the Riesz fractional derivative. *J. Comput. Phys.*, 231(4):1743–1750, 2012. doi:10.1016/j.jcp.2011.11.008.

G. Ceruti, J. Kusch, and C. Lubich. A rank-adaptive robust integrator for dynamical low-rank approximation. *BIT*, pages 1–26, 2022. doi:10.1007/s10543-021-00907-7.

G. Ceruti and C. Lubich. Time integration of symmetric and anti-symmetric low-rank matrices and Tucker tensors. *BIT*, 60(3):591–614, 2020. doi:10.1007/s10543-019-00799-8.

G. Ceruti and C. Lubich. An unconventional robust integrator for dynamical low-rank approximation. *BIT*, pages 591–614, 2021. doi:10.1007/s10543-021-00873-0. Online first.

E. M. Constantinescu. Generalizing global error estimation for ordinary differential equations by using coupled time-stepping methods. *J. Comput. Appl. Math.*, 332:140–158, 2018. doi:10.1016/j.cam.2017.05.012.

A. Dektor, A. Rodgers, and D. Venturi. Rank-adaptive tensor methods for high-dimensional nonlinear PDEs. *J. Sci. Comput.*, 88(2):Paper No. 36, 27, 2021. doi:10.1007/s10915-021-01539-3.

P. Deuflhard. A study of extrapolation methods based on multistep schemes without parasitic solutions. *Z. Angew. Math. Phys.*, 30(2):177–189, 1979. doi:10.1007/BF01601932.

M. Eiermann, O. G. Ernst, and S. Güttel. Deflated restarting for matrix functions. *SIAM J. Matrix Anal. Appl.*, 32(2):621–641, 2011. doi:10.1137/090774665.

L. Einkemmer and C. Lubich. A low-rank projector-splitting integrator for the vlasov–poisson equation. *SIAM Journal on Scientific Computing*, 40(5):B1330–B1360, 2018. doi:10.1137/18M116383X.

L. Einkemmer, A. Ostermann, and C. Piazzola. A low-rank projector-splitting integrator for the vlasov–maxwell equations with divergence correction. *Journal of Computational Physics*, 403:109063, 2020. doi:10.1016/j.jcp.2019.109063.

W. Gautschi. Numerical integration of ordinary differential equations based on trigonometric polynomials. *Numer. Math.*, 3:381–397, 1961. doi:10.1007/BF01386037.

T. Göckler. *Rational Krylov Subspace Methods for phi-Functions in Exponential Integrators*. Dissertation, Karlsruhe Institute of Technology (KIT), 2014. doi:10.5445/IR/1000043647.

G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

V. Grimm and M. Hochbruck. Error analysis of exponential integrators for oscillatory second-order differential equations. *J. Phys. A*, 39(19):5495–5507, 2006. doi:10.1088/0305-4470/39/19/S10.

E. Hairer, C. Lubich, and G. Wanner. Geometric numerical integration illustrated by the störmer–verlet method. *Acta Numerica*, 12:399–450, 2003. doi:10.1017/S0962492902000144.

E. Hairer, C. Lubich, and G. Wanner. *Geometric numerical integration : structure-preserving algorithms for ordinary differential equations*. Springer series in computational mathematics ; 31. Springer, Berlin, 2. ed. edition, 2006. doi:10.1007/3-540-30666-8.

E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations I: Nonstiff problems*, volume 8 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 1993. doi:10.1007/978-3-540-78862-1.

E. Hairer and G. Wanner. *Solving ordinary differential equations. II*, volume 14 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 2010. doi:10.1007/978-3-642-05221-7. Stiff and differential-algebraic problems, Second revised edition, paperback.

B. C. Hall. *Lie groups, Lie algebras, and representations*, volume 222 of *Graduate Texts in Mathematics*. Springer, Cham, second edition, 2015. doi:10.1007/978-3-319-13467-3. An elementary introduction.

H. V. Henderson and S. R. Searle. The vec-permutation matrix, the vec operator and Kronecker products: a review. *Linear and Multilinear Algebra*, 9(4):271–288, 1980/81. doi:10.1080/03081088108817379.

J. S. Hesthaven, C. Pagliantini, and N. Ripamonti. Rank-adaptive structure-preserving model order reduction of Hamiltonian systems. *ESAIM Math. Model. Numer. Anal.*, 56(2):617–650, 2022. doi:10.1051/m2an/2022013.

N. J. Higham. *Functions of matrices: Theory and computation*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008. doi:10.1137/1.9780898717778.

M. Hochbruck and C. Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 34(5):1911–1925, 1997. doi:10.1137/S0036142995280572.

M. Hochbruck, M. Neher, and S. Schrammer. Dynamical low-rank integrators for second-order matrix differential equations. CRC 1173 Preprint 2022/12, Karlsruhe Institute of Technology, 2022a. doi:10.5445/IR/1000143003.

M. Hochbruck, M. Neher, and S. Schrammer. Rank-adaptive dynamical low-rank integrators for first-order and second-order matrix differential equations. CRC 1173 Preprint 2022/13, Karlsruhe Institute of Technology, 2022b. doi:10.5445/IR/1000143198.

M. Hochbruck, T. Pažur, A. Schulz, E. Thawinan, and C. Wieners. Efficient time integration for discontinuous Galerkin approximations of linear wave equations [Plenary lecture presented at the 83rd Annual GAMM Conference, Darmstadt, 26th–30th March, 2012]. *ZAMM Z. Angew. Math. Mech.*, 95(3):237–259, 2015. doi:10.1002/zamm.201300306.

R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, Cambridge, 1990. Corrected reprint of the 1985 original.

R. A. Horn and C. R. Johnson. *Topics in matrix analysis*. Cambridge University Press, Cambridge, 1994. Corrected reprint of the 1991 original.

J. Hu and Y. Wang. An adaptive dynamical low rank method for the nonlinear boltzmann equation, 2021. URL `https://arxiv.org/abs/2112.02695`.

C. Karle, J. Schweitzer, M. Hochbruck, E. W. Laedke, and K. H. Spatschek. Numerical solution of nonlinear wave equations in stratified dispersive media. *J. Comput. Phys.*, 216(1):138–152, 2006. doi:10.1016/j.jcp.2005.11.024.

C. Karle, J. Schweitzer, M. Hochbruck, and K. H. Spatschek. A parallel implementation of a two-dimensional fluid laser-plasma integrator for stratified plasma-vacuum systems. *J. Comput. Phys.*, 227(16):7701–7719, 2008. doi:10.1016/j.jcp.2008.04.024.

E. Kieri, C. Lubich, and H. Walach. Discretized dynamical low-rank approximation in the presence of small singular values. *SIAM J. Numer. Anal.*, 54(2):1020–1038, 2016. doi:10.1137/15M1026791.

O. Koch and C. Lubich. Dynamical low-rank approximation. *SIAM Journal on Matrix Analysis and Applications*, 29(2): 434–454, 2007. doi:10.1137/050639703.

J. Kusch, G. Ceruti, L. Einkemmer, and M. Frank. Dynamical low-rank approximation for Burgers' equation with uncertainty. *Int. J. Uncertain. Quantif.*, pages 1–24, 2022. doi:10.1615/Int.J.UncertaintyQuantification.2022039345. Online first.

J. Kusch and P. Stammer. A robust collision source method for rank adaptive dynamical low-rank approximation in radiation therapy. CRC 1173 Preprint 2022/5, Karlsruhe Institute of Technology, 2022. doi:10.5445/IR/1000141755.

J. M. Lee. *Introduction to Smooth Manifolds*. Graduate Texts in Mathematics. Springer, New York, NY, 2nd edition, 2012. doi:10.1007/978-1-4419-9982-5.

S. T. Lee, H.-K. Pang, and H.-W. Sun. Shift-invert Arnoldi approximation to the Toeplitz matrix exponential. *SIAM J. Sci. Comput.*, 32(2):774–792, 2010. doi:10.1137/090758064.

C. Lubich and I. Oseledets. A projector-splitting integrator for dynamical low-rank approximation. *BIT Numerical Mathematics*, 54(1):171–188, 2014. doi:10.1007/s10543-013-0454-0.

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

A. Ruhe. Rational Krylov sequence methods for eigenvalue computation. *Linear Algebra Appl.*, 58:391–405, 1984. doi:10.1016/0024-3795(84)90221-0.

S. Schrammer. Codes for numerical experiments, 2022. doi:10.5445/IR/1000148858.

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

G. Strang. The discrete cosine transform. *SIAM Rev.*, 41(1):135–147, 1999. doi:10.1137/S0036144598336745.

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

V. E. Tarasov and G. M. Zaslavsky. Fractional ginzburg–landau equation for fractal media. *Physica A: Statistical Mechanics and its Applications*, 354:249–261, 2005. doi:10.1016/j.physa.2005.02.047.

L. N. Trefethen and D. Bau, III. *Numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997. doi:10.1137/1.9780898719574.

C. F. Van Loan. The ubiquitous kronecker product. *Journal of Computational and Applied Mathematics*, 123(1):85–100, 2000. doi:10.1016/S0377-0427(00)00393-9. Numerical Analysis 2000. Vol. III: Linear Algebra.

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

X. Zhao, Z.-z. Sun, and Z.-p. Hao. A fourth-order compact adi scheme for two-dimensional nonlinear space fractional schrödinger equation. *SIAM Journal on Scientific Computing*, 36(6):A2865–A2886, 2014. doi:10.1137/140961560.

Y.-L. Zhao, A. Ostermann, and X.-M. Gu. A low-rank Lie-Trotter splitting approach for nonlinear fractional complex Ginzburg-Landau equations. *J. Comput. Phys.*, 446:Paper No. 110652, 12, 2021. doi:10.1016/j.jcp.2021.110652.

Z. Zlatev, I. Dimov, I. Faragó, and Ágnes Havasi. *Richardson Extrapolation: Practical Aspects and Applications*. De Gruyter, 2017. doi:10.1515/9783110533002.