Time integration for the dynamical low-rank approximation of matrices and tensors

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat)

vorgelegt von

Dipl.-Math. Hanna Maria Walach

aus Rybnik (Polen)

Tübingen
2018

Tag der mündlichen Qualifikation:	24.05.2019
Dekan:	Prof. Dr. Wolfgang Rosenstiel
1. Berichterstatter:	Prof. Dr. Christian Lubich
2. Berichterstatter:	Prof. Dr. André Uschmajew

Für meine Familie und für David

Abstract

This thesis is concerned with the low-rank approximation of time-dependent high-dimensional matrices and tensors that can be given explicitly or are the unknown solution to a matrix or tensor differential equation. Large differential equations typically arise from a space discretization of a high-dimensional evolutionary partial differential equation and are not solvable by direct discretization because of their sheer size. The dynamical low-rank approximation approach counters this computational infeasibility by evolving a differential equation for an approximation matrix or tensor with underlying low-rank structure. The Lipschitz constant of the right-hand side of this differential equation grows inversely proportional to the size of the smallest singular value of the approximation matrix or of matricizations of the approximate tensor. Therefore, standard numerical integrators deteriorate in this situation. In practice, small singular values appear often due to overapproximation.

A constitutive method for time integration of matrices in low-rank format is the matrix projector-splitting integrator. It updates factor matrices of the underlying truncated singular value decomposition. We present a rigorous error analysis for this integrator that shows its robustness with respect to small singular values and its first order convergence. This result is achieved by using the exactness property of the integrator and the preservation of subspaces during the integration procedure. By means of the same ingredients, we extend this error analysis to the time integrator of tensor trains.

We further derive an integration method for time-dependent Tucker tensors. Matricizations of Tucker tensors enable us to a nested application of a modified version of the matrix projector-splitting integrator, where a substep in the integration step is not done exactly, but by another low-rank approximation. This nested Tucker integrator turns out to be exact in the explicit case and robust in the presence of small singular values of matricizations of the Tucker tensor.

We also propose a numerical integrator for the approximation of a matrix that is the unknown solution to a stiff matrix differential equation. We deal with a class of matrix differential equations that is characterized by a stiff linear and a non-stiff nonlinear part. This integrator separates the stiff differential equation into a linear and a nonlinear subproblem by the Lie–Trotter splitting method. We show an error bound of order one that is independent of singular values and of the severe Lipschitz constant.

We contribute to the development and to the analysis of efficient and robust time-integration methods by following the dynamical low-rank approximation approach using low-rank structures of matrix and tensor representations.

Zusammenfassung

Die Niedrigrangapproximation zeitabhängiger, hochdimensionaler Matrizen und Tensoren, die explizit oder implizit als unbekannte Lösung einer Matrix- oder Tensordifferentialgleichung gegeben sein können, ist Gegenstand der Betrachtung. Differentialgleichungen für sehr große Matrizen und Tensoren treten typischerweise nach der Ortsdiskretisierung einer hochdimensionalen partiellen Differentialgleichung auf und sind auf Grund der Größe der Matrix beziehungsweise des Tensors nicht direkt lösbar. Der Ansatz der dynamischen Niedrigrangapproximation bringt eine Differentialgleichung für die Approximationsmatrix oder den -tensor mit Niedrigrangstruktur hervor und wirkt den rechentechnischen Schwierigkeiten auf diese Weise entgegen. Die Lipschitzkonstante der rechten Seite dieser Differentialgleichung verhält sich jedoch proportional zur Inversen des kleinsten Singulärwertes der Approximationsmatrix beziehungsweise der Matrizisierungen des Approximationstensors. Aus diesem Grund sind klassische numerische Verfahren nicht praktikabel, da sie eine starke Schrittweitenbeschränkung erfordern, um Lösungen zu liefern. In der Anwendung der Niedrigrangapproximation ist a priori nicht klar wie groß der effektive Rang der zu approximierenden Matrix oder des Tensors ist und daher wird dieser oft zu groß gewählt. Dies führt dazu, dass kleine Singulärwerte auftreten.

Der Matrixintegrator ist ein wesentliches Verfahren für die Zeitintegration von Matrizen im Singulärwert zerlegten Niedrigrangformat und ist grundlegend für diese Arbeit. Er bestimmt die drei Faktormatrizen zum nächsten Zeitpunkt und liefert so eine Approximationslösung von niedrigem Rang. Wir führen eine Fehleranalyse dieses Integrators durch, die eine Konvergenz erster Ordnung zeigt und die eine Fehlerschranke unabhängig von kleinen Singulärwerten nachweist. Um die Schwierigkeit mit der Lipschitzkonstante zu umgehen, machen wir Gebrauch von der Exaktheit des Integrators im expliziten Fall und von der Beobachtung, dass jeweils eine der beiden Basismatrizen der Singulärwertzerlegung während der Zeitintegration konstant bleibt. Mit den gleichen Ideen lässt sich die Fehleranalyse für den Integrator für Tensor Trains ausweiten.

Ferner entwickeln wir eine Integrationsmethode für die Zeitentwicklung von Tucker-Tensoren. Die Matrizisierung von Tucker Tensoren erlaubt es uns eine leicht abgeänderte Version des Matrixintegrators anzuwenden, indem wir die ersten beiden Teilschritte direkt lösen, beim dritten Schritt hingegen eine Niedrigrangapproximation durchführen. Dieser Tucker Integrator ist exakt wenn der zu approximierende Tensor explizit gegeben ist. Dieses Verfahren liefert auch bei auftretenden kleinen Singulärwerten gute Ergebnisse, was aus der Fehleranalyse hervorgeht, die Fehlerschranken angibt, welche unabhängig von Singulärwerten sind.

Überdies beschäftigen wir uns mit der Niedrigrangapproximation von Lösungsmatrizen steifer Differentialgleichungen. Hierbei betrachten wir jene Differentialgleichungen, die aus einem linearen und steifen sowie einem nichtlinearen und nicht steifen Anteil bestehen. Die Hauptidee dieses Integrationsverfahrens besteht darin, den steifen vom nicht steifen Anteil mit Hilfe der Lie–Trotter Splittingmethode zu trennen und die beiden resultierenden Differentialgleichungen für sich zu lösen. Auf Grund dieser Aufteilung ist es möglich eine Fehleranalyse zu führen, die aufzeigt, dass das Verfahren von der Lipschitzkonstanten nicht beeinflusst wird und dass dessen Fehlerschranke unabhängig von Singulärwerten ist.

Die vorliegende Arbeit ist ein Beitrag zur Entwicklung sowie zur numerischen Analyse effizienter und bezüglich kleiner Singulärwerte robuster numerischer Integrationsverfahren. Grundlegend hierfür ist das Verfahren der dynamischen Niedrigrangapproximation unter Verwendung einer Niedrigrangfaktorisierung der Matrix oder des Tensors.

Contributions and sources

We give an overview over the new results and their origins on which this thesis is mainly based.

Chapter 2 is concerned with the error analysis of the matrix projector-splitting integrator. Its error analysis given in Section 2.3, the error bounds for specific situations given in Section 2.4 and the error estimate of the time-integration method for tensor trains in Section 5.1 originate from a collaboration of the author with E. Kieri and Ch. Lubich and are published in [KLW16]. The contributions on the theoretical results in this work are equally distributed amongst the authors. The numerical examples in Section 2.5, also taken from [KLW16], are due to E. Kieri and the author, where both contributed equally: the examples about the discrete nonlinear, and about the stiff Schrödinger equation, see Sections 2.5.3 and 2.5.4, are implemented by E. Kieri and the experiments about the matrix addition in Section 2.5.2, the tensor addition mentioned in Section 5.1 as well as the example in Section 2.5.1 are coded by the author. The proof given in Section 2.2.1 about the exactness of the projector-splitting integrator is solely due to the author and has not been published or submitted elsewhere.

The low-rank splitting integrator and its analysis presented in Chapter 3 are entirely drawn from [OPW18], a collaborative work of the author with A. Ostermann and C. Piazzola. This manuscript is unpublished, yet submitted. The scientific ideas in Sections 3.1-3.5 are portioned out equally among the three authors, except of the algorithmic description of the integrator in Section 3.1.3, which is due to the author. The numerical examples in Section 3.6 are implemented by C. Piazzola.

The new results presented in Chapter 4 have its source in [LVW18], which originates from a cooperation of the author with Ch. Lubich and B. Vandereycken. The three authors contributed equally to the theoretical results in Sections 4.2-4.5 and in Section 4.6.5, whereas the numerical experiments presented in Section 4.7 are implemented by the author. The direct exactness proof of the projector-splitting Tucker integrator given in Section 4.6.3 is a result due to the author and is neither published, nor submitted.

In all chapters, where we have presented results from the above sources, we have rather freely adapted them: we have provided more details in the theoretical results and we have standardized the notation.

Danke

Ich möchte diese Gelegenheit nutzen und mich bei allen, die fachlich sowie außerfachlich zum Entstehen dieser Arbeit beigetragen haben, bedanken.

Mein ganz besonderer Dank gilt Prof. Dr. Christian Lubich. Danke für die Möglichkeit zur Forschung, für die exzellente Betreuung und sein außerordentliches Engagement, für die vielen Antworten auf meine Fragen, für die ständige Bereitschaft zur Diskussion und das Teilen von Ideen sowie für das Ermöglichen der vielen Reisen, die ich unternehmen durfte. Christian, ich habe viel von dir gelernt.

Während meines Forschungsaufenthaltes in Genf habe ich viele Menschen getroffen, bei denen ich mich bedanken möchte.
Vielen Dank an Prof. Dr. Bart Vandereycken, der mich in seiner Arbeitsgruppe aufgenommen hat. Bart, danke für die produktive Zusammenarbeit und auch für unseren außerfachlichen Austausch.
Danke an die gesamte Arbeitsgruppe der Numeriker in Genf. Ihr habt es mir sehr leicht gemacht mich bei euch wohl zu fühlen.
Mein großer Dank gilt Gerhard und Myriam Wanner, bei denen ich zu Hause sein und mit denen ich unzählige Ausflüge unternehmen sowie einige Rommé-Abende erleben durfte. Ich hatte eine prima Zeit mit euch und danke euch außerdem für die gemütliche Hängematte im Wohnzimmer.
Die Zeit in Genf hat meinen fachlichen wie auch persönlichen Horizont erweitert und wird mir in guter Erinnerung bleiben.

Mein Dank gilt Prof. Dr. Alexander Ostermann und Chiara Piazzola für die gute Zusammenarbeit, wir haben uns thematisch sehr gut ergänzt. Außerdem danke ich Chiara für die angenehmen persönlichen Gespräche über das Doktorandendasein.

Diese Arbeit sowie meine Konferenzteilnahmen wurden aus finanziellen Mitteln der DFG im Projekt GRK1838 gefördert, wofür ich zu Dank verpflichtet bin.

Die gute (Arbeits)-atmosphäre, die vielen lebhaften Kaffeerunden und die abwechslungsreichen außeruniversitären Unternehmungen mit den Tübinger Numerikern waren mir eine sehr willkommene Ablenkung von meiner Forschung. Dafür danke ich Bernd Brumm, meinem room mate Gianluca Ceruti - es war mir eine Freude mit dir über dynamical low-rank approximation und „let Y be a tensor“ zu diskutieren, Sarah Eberle, Dominik Edelmann, Dhia Mansour, Jörg Nick, Christian Power und nicht zuletzt Jonathan Seyrich.

Ausdrücklich bedanken möchte ich mich bei Balázs Kovács. Unsere tagtäglichen „morning chats“ waren einfach legendär, sie haben uns einen guten Start in den Arbeitstag beschert. Balázs, danke, dass deine Türe mir für fachliche und insbesondere außerfachliche Themen immer offen stand. Und danke auch für das sorgfältige Durchlesen dieser Arbeit.
Unsere Gruppendynamik wurde zeitweise durch Lukas Einkemmer, Emil Kieri, dem ich insbesondere für unsere gelungene Zusammenarbeit danke, Buyang Li sowie Bin Wang bereichert.

Mein tiefer und herzlicher Dank gilt meiner Familie und insbesondere meinen Eltern. Danke für eure vielfältige und immerwährende Unterstützung und für euren guten Rat.
Meinem Freund David danke ich für seinen Rückhalt. David, danke, dass du mich immer wieder in meinem Vorhaben bestärkt, mich motiviert sowie meine Höhen mit Freude und meine Tiefen mit Gelassenheit während dieser ganzen Zeit begleitet hast.
Obwohl wir uns thematisch nicht austauschen konnten, habt ihr zweifelsfrei zum Gelingen dieser Dissertation beigetragen. Danke!

Contents

Introduction

Brain waves move a wheelchair in real time. This surreal sounding statement is fortunately a matter of fact. In the fields of biomedicine, bioengineering and neuroscience, the area of brain machine interface has gained great success in this development in the past ten years. Such systems help elderly and handicapped people to live autonomously through signals from their minds. The computer installed in the wheelchair scans brain waves through electroencephalography (EEG). In 125 milliseconds, the computer turns a thought into a command to move the wheelchair forward or turn it left or right, such that it moves smoothly due to real time control. EEG signals are widely used to measure brain waves in neuroscience, such as also for an early diagnosis of Alzheimer's disease, and they are usually stored in a multidimensional array, say $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ at each time t. In such complex applications, the amount of data is very large, here we would have to store about N^d entries, where $N = \max\{n_1, \ldots, n_d\}$, which means an exponential growth in dimension d with regard to memory requirements. For processing the data set in order to move the wheelchair, the computer has to work on each data point to gain information about passing the way the person intends to move. In case of a large data set $A(t)$ the data processing takes a prohibitively long time, such that the person does not move from one place to another in real time, but for instance after several hours, when the computations would be finished.

To put such a helpful device into practice, the challenge is to drastically reduce the measured data up to the most necessary, without losing too much information, since otherwise the computer would run out of time and memory, and a real time control over the wheelchair would therefore not be possible. There is rich literature about these developments, and here we only list a selection [CC08, CWR$^+$08, CZPA09, Cic13] and the review article [CLK$^+$15].

From a mathematical point of view, our aim is to approximate the time-dependent array $A(t)$ by another array $Y(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, i.e.,

$$A(t) \approx Y(t),$$

where the storage of the approximation array as well as its processing is computationally less expensive than working with $A(t)$.

Let us now turn from brain waves to quantum state waves. In natural sciences and in engineering most of the phenomena are not described explicitly by data stored in an array, but are rather given implicitly as the unknown solution to a high-dimensional ordinary or partial differential equation. In the field of quantum mechanics, for instance, the state of

a molecular system at time t is described by the wave function ψ, which is given as the solution to the linear time-dependent Schrödinger equation

$$i\frac{\partial \psi}{\partial t}(x,t) = H(x,t)\psi(x,t), \qquad x \in \mathbb{R}^d,\ t \geq 0,$$

with a Hamiltonian operator H for a multi-particle wave function $\psi(x,t) = \psi(x_1 \ldots, x_d, t)$, which is a partial differential equation, see [Sch26, Lub08] and references therein. Many interesting models in quantum mechanics involve several, say d, particles, such that the dimension d might become large. The computational challenges with regard to time and memory when solving such high-dimensional partial differential equations are very demanding. In fact, they evoke from the high dimension of the considered problem and this computational intractability is termed by the catchphrase "curse of dimensionality", see [Bel61]. Already back in 1930, P.A.M. Dirac realized by studying the Thomas atom in [Dir30a] that "for dealing with atoms involving many electrons the accurate quantum theory, involving a solution of the wave equation in many-dimensional space, is far too complicated to be practicable. One must therefore resort to approximate methods."

Following his suggestion, from the numerical analysis point of view, we discretize a partial differential equation in space and obtain a system of ordinary differential equations of the form

$$\dot{A}(t) = F(t, A(t)), \qquad A(t_0) = A^0,$$

where $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ for all times $t_0 \leq t \leq T$. This type of differential equation, which is characterized by a high dimension of the solution $A(t)$, marks the starting point of this thesis. Such a differential equation is not directly tractable because of its sheer size, except for very small dimensions d.

In the same work, Dirac also proposed an ansatz to find approximate solutions to high-dimensional problems, such as the Schrödinger equation, which nowadays is known as the *Dirac–Frenkel variational principle* in quantum mechanics literature, see [Dir30a, Dir30b, Fre34], and which will play a fundamental role in this thesis. The main idea is to evolve a differential equation for the approximation array $Y(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. Looking at the dimension of the approximation array, it is reasonable to ask about the computational benefit when solving a differential equation for $Y(t)$ instead of solving a differential equation for $A(t)$. The solution of the differential equation for the approximation $Y(t)$ becomes computationally accessible if it admits a low-rank structure, which will be described in a later paragraph.

Throughout this thesis, we aim to find an approximation $Y(t)$ for all $t_0 \leq t \leq T$, characterized by arrays of the same size as $A(t)$ that admit a (low) rank r structure, where we will specify r later, and we identify this search space as a low-rank manifold $\mathcal{M}$. In the numerical analysis literature, the problem of finding a low-rank approximation of an explicitly or implicitly given time-dependent array and its numerical treatment were first studied by O. Koch and Ch. Lubich in [KL07] for the two-dimensional matrix case. There, the ansatz of the *dynamical low-rank approximation* is introduced, which mainly follows

the ideas of the Dirac–Frenkel variational principle: the right-hand side of the differential equation for $A(t)$ is projected orthogonally onto the tangent space of the low-rank manifold at an approximation $Y(t)$, for all $t_0 \leq t \leq T$. Denoting the orthogonal projection at $Y(t)$ by $\mathrm{P}(Y(t))$, this results in a differential equation

$$\dot{Y}(t) = \mathrm{P}(Y(t))F(t, Y(t)), \qquad Y(t_0) = Y^0$$

for the approximation array, which due to the underlying low-rank structure of $Y(t)$ leads to a reduced model compared to the differential equation for $A(t)$. Employing and exploiting the low-rank representation of the approximation array while integrating its differential equation is the key to computational efficiency of the dynamical low-rank approximation approach. A first numerical implementation and some applications are reported in [NL08]. We also refer to the review article [Lub14] about low-rank dynamics.

We shall now introduce the low-rank formats that are central for this thesis to overcome the curse of dimensionality to some extent.

For matrices, the question about finding an approximation matrix to an explicitly given matrix can be traced back to C.F. Gauss in 1809, though having a different motivation to this problem, see [Gau09, Gau23]. About one hundred years later, a group of mathematicians contributed to finding an approximation matrix $\mathbf{Y}(t) \in \mathbb{R}^{n_1 \times n_2}$ to the given matrix $\mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_2}$ as best as possible, which nowadays is known as the *singular value decomposition*. There are several proofs showing that every matrix admits a singular value decomposition, where the most prominent proof is by construction, see, for instance, in the monograph [GVL96]. This non-unique factorization decomposes the matrix $\mathbf{A}(t)$ into three matrices.

In order to determine an approximation matrix of (low) rank r, we zero out the last $(N-r)$ singular values of $\mathbf{A}(t)$, which results in the *truncated* singular value decomposition

$$\mathbf{Y}(t) = \mathbf{U}(t)\mathbf{S}(t)\,\mathbf{V}(t)^\top,$$

where $\mathbf{U}(t) \in \mathbb{R}^{n_1 \times r}$, $\mathbf{V}(t) \in \mathbb{R}^{n_2 \times r}$ have orthonormal columns, $\mathbf{S}(t) \in \mathbb{R}^{r \times r}$ and $r < \min\{n_1, n_2\}$. Of course, in general, the matrix $\mathbf{A}(t)$ will have a singular value decomposition at each time $t \in [t_0, T]$, but we think of those factor matrices to be smooth, since this is desired in the later applied time integration methods. As for smooth decompositions of matrices, see [DE99].

Now, imposing the rank r of the approximation matrix $\mathbf{Y}(t)$ to be much smaller than the rank of $\mathbf{A}(t)$, the singular value factorization can be stored and manipulated more economically than the matrix $\mathbf{A}(t)$ itself. Supposing that $N = \max\{n_1, n_2\}$, storing the full matrix $\mathbf{Y}(t)$ requires saving about N^2 entries. In contrast, employing the low-rank decomposition, we simply store the factor matrices, which accounts for a memory requirement of $2Nr + r^2$ entries. In case when $r \ll \min\{n_1, n_2\}$, this leads to a significant reduction of computational cost.

There is a rich literature about the genesis and early developments of the singular value decomposition, such as [Bel73, Jor74, Sch07, EY36, Mir60, GVL96], while see [Ste93] for an overview on the early history of the singular value decomposition.

Computing a singular value decomposition of a matrix is a task for its own. There are several different approaches of how to determine this factorization from the computational viewpoint. The most prominent algorithm proposed in [GK65] consists of two phases based on QR decompositions: first, the given matrix is transformed in a bidiagonal form and second, the bidiagonal form is diagonalized and its singular values are computed. There are many other methods, which follow the first step of this algorithm, but modify the second part, such as the one- or two sided Jacobi algorithm [GVL96], the modified QR iteration [DK90] for computing singular values with high relative accuracy or the method proposed in [TB97] that is based on divide-and-conquer eigenvalue algorithms. In this thesis, we use the truncated singular value decomposition as a mathematical tool and identify it as a low-rank representation of two-dimensional arrays.

Increasing the dimension of the array from two to $d \geq 3$, we speak about tensors. The first and most basic idea to decompose a tensor to a low-rank representation is about reducing it to a sum of r tensor products of vectors in d dimensions. This tensor format is known as canonical polyadic decomposition and is introduced in [Hit27], but did not find attention in the mathematical community. It is redeveloped in [Har70] and in [CC70] and renamed as PARAFAC/CANDECOMP in the psychometrics and chemometrics literature, where it gained recognition. In case when $N = \max\{n_1, \ldots, n_d\}$, it requires only dNr entries to be stored and so the dependence on the dimension d of the tensor is only linear. Computing the canonical format out of a given tensor is a drawback of this representation, since the existing methods are unstable, and so this tensor representation did not find broad attention in the mathematical community.

In the mid 20th century, tensor decompositions were taken up by the development of the fundamental *Tucker tensor format*. It was introduced by L.R. Tucker for three dimensional tensors in the psychometrics literature in [Tuc66] for the principal component analysis. Since then, it has seen various applications and its usage spread out to the numerical linear algebra and to the numerical analysis community. The authors in [DLDMV00a] have shown that every high-dimensional tensor admits an exact Tucker representation. The Tucker decomposition is a natural generalization of the singular value factorization from the matrix to the tensor case. It decomposes a tensor exactly into a core tensor, which is of the same size as the tensor itself, multiplied by matrices along each mode $i = 1, \ldots, d$. With regard to computational effort, this decomposition is inadequate in this form and requires a modification of the factors. Similarly to the matrix case, the last $(n_i - r_i)$ entries of the core tensor and of the matrices are truncated in each mode $i = 1, \ldots, d$, which results in the *truncated* Tucker decomposition

$$Y(k_1, \ldots, k_d)(t) = \sum_{j_1=1}^{r_1} \cdots \sum_{j_d=1}^{r_d} C(j_1, \ldots, j_d)(t) \cdot \mathbf{U}_1(k_1, j_1)(t) \cdots \mathbf{U}_d(k_d, j_d)(t),$$

where $C(j_1, \ldots, j_d)(t)$ are the entries of the core tensor $C(t) \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ and $\mathbf{U}_i(k_i, j_i)(t)$ are the elements of the factor matrices $\mathbf{U}_i(t) \in \mathbb{R}^{n_i \times r_i}$, for all $i = 1, \ldots, d$ and for all $t \in [t_0, T]$.

Tucker also presented a method of how to compute this tensor decomposition in [Tuc66, Method 1]. This fragmented algorithm is replaced by the truncated higher-order singular

value decomposition (where the notion *order* is used in place of the term *dimension* and has the same meaning), a stable method based on truncating singular values of matricizations of the given tensor in each mode, introduced in [DLDMV00a], and enhanced as the higher order orthogonal iteration method in [DLDMV00b]. In contrast to the matrix case, the truncated higher-order singular value decomposition does not yield a best-approximation, but it results in a quasi-optimal approximation tensor of multilinear rank [Hit28]. From the computational perspective, the Tucker format requires the storage of about $R^d + dNR$ entries, where $R = \max\{r_1, \ldots, r_d\}$. Supposing $R \ll N$, this is a great reduction of computational memory compared to storing the full tensor with its N^d entries. However, its storage scales exponentially with the rank, which makes the Tucker format suitable for the three- or four-dimensional case.

A remedy for this exponential scaling is the development of the *tensor train decomposition* [OT09, Ose11]. In the theoretical physics literature, this format was already known as matrix product states, see, e.g., [PGVWC06, VMC08, Sch11, HOV13, HLO$^+$16]. The main idea of the tensor train representation is to approximate a tensor of high order d with a collection of tensors of order three. Each entry of a tensor in the tensor train format admits the form

$$Y(k_1, \ldots, k_d)(t) = \sum_{j_1=1}^{r_1} \cdots \sum_{j_{d-1}=1}^{r_{d-1}} C_1(1, k_1, j_1)(t) \cdot C_2(j_1, k_2, j_2)(t) \cdots C_d(j_{d-1}, k_d, 1)(t),$$

where the three-dimensional core tensors are of size $C_i(t) \in \mathbb{R}^{r_{i-1} \times n_i \times r_i}$ for each mode $i = 1, \ldots, d$, with $r_0 = r_d = 1$ and for all $t \in [t_0, T]$. This decomposition can be obtained by the TT-SVD algorithm proposed in [Ose11], which is based on truncated singular value decompositions of matricizations of the core tensors. It yields a quasi best approximation to a full higher-order tensor and is the state-of-the-art method to compute the tensor train factorization. In contrast to the Tucker tensor format, it scales only quadratic with the rank, since the computational memory amounts for storing about $(d-1)NR^2$ entries.

For an introduction to further tensor formats and tensor calculus, we refer to [Hac12].

This thesis contributes to the dynamical low-rank approximation of matrices, Tucker tensors and tensor trains, where all are of fixed rank, but change in time. As mentioned earlier, when following the dynamical low-rank approximation, we are interested in solving the reduced differential equation for the approximation array $Y(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ in an efficient way by benefiting from an underlying low-rank representation format.

For two-dimensional arrays $\mathbf{Y}(t)$, the first integration method was proposed in [KL07], where a system of differential equations for the factors $\mathbf{U}(t)$, $\mathbf{S}(t)$ and $\mathbf{V}(t)$ of the singular value decomposed format were determined from the above specified differential equation. Solving those differential equations by a standard numerical integration method, like Runge–Kutta as suggested in [KL07], leads to an update of the factor matrices and by building their product to the desired update of the approximation matrix $\mathbf{Y}(t)$ from one time step to the next.

This numerical integration method yields locally quasi-optimal approximation results and the local error bound depends on the inverse of the smallest singular value of the

approximation matrix $\mathbf{Y}(t)$. So in case when the smallest singular value of $\mathbf{Y}(t)$ is small, this error bound is unfeasible. The proof of this integration method uses a severe property of the orthogonal projection: the local Lipschitz constant of the orthogonal projection at $\mathbf{Y}(t)$ is inversely proportional to the smallest non-zero singular value of the approximation matrix and can thus become arbitrarily large, see [KL07, Lemma 4.2].

From a geometric viewpoint regarding the low-rank manifold this means that the curvature of the manifold is high for matrices with singular values that are close to zero. From the numerical analysis point of view, standard error estimates break down in the presence of small singular value, since they use the local Lipschitz constant of the tangent space projection.

Another difficulty this property brings about affects the numerical integration of the system of differential equations. Due to the usage of the singular value factorization as the low-rank representing matrix format, the singular values of $\mathbf{Y}(t)$ are contained in the matrix $\mathbf{S}(t)$, whose inverse appears on the right-hand side of the differental equations for $\mathbf{U}(t)$ and $\mathbf{V}(t)$, respectively. If the singular values of the approximation matrix are small, this leads to a severe step size restriction when applying standard numerical integrators, such as Runge–Kutta methods.

The difficulties with small singular values are a restrictive observation and therefore it is reasonable to ask whether one has to expect occuring small singular values in practice, or whether this is a problem of rather theoretical nature. In fact, this question is linked to the problem of how to choose the approximation rank. A rough choice for the approximation rank suggests to choose $r \approx \log(N)$, where $N = \max\{n_1, \ldots, n_d\}$, see [KK18], but this hypothesis is only approved by numerical experiments, yet not proven rigorously. Even if there is a distinct gap in the singular value distribution such that two groups of large and negligibly small singular values, respectively, are formed, it is typically not known a priori at which rank the group of effective singular values ends. Choosing the approximation rank too small, we run the risk of neglecting singular values that are large and account for the effective rank, such that underestimating the rank saves computational effort, but leads to loss of information. On the other hand, picking r too large means taking small singular values under consideration, which implicates problems with large Lipschitz constants. Without having any knowledge of the effective rank of the approximation matrix, the choice of the approximation rank is accidental. Typically one chooses the rank rather large in order to avoid losing information, and so, being faced with appearing small singular values in $\mathbf{Y}(t)$ is very common.

Recently, very efficient integrators based on splitting the projection onto the tangent space of the low-rank manifold have been proposed and studied for matrices [LO14] and for Tucker tensors [Lub15] as well as for tensor trains [LOV15].

Let us give a more detailed introduction into those three integration schemes, since they build the starting point for the developments of this dissertation. We begin with the two-dimensional matrix integrator. As mentioned earlier, the first integration scheme for matrices proposed in [KL07] deteriorates in the presence of small singular values. In contrast, the *matrix projector-splitting integrator* of [LO14] performs well in the case of an ill-conditioned $\mathbf{S}(t)$, because it computes updates of the factors $\mathbf{U}(t)$, $\mathbf{S}(t)$ and $\mathbf{V}(t)$,

respectively, in a different way. This favorable behavior was observed in numerical examples and shown analytically in the special case of a distinct gap in the distribution of the singular values of given time-dependent matrices $\mathbf{A}(t)$ in [LO14]. Another remarkable characteristic is the *exactness property* of the integrator: it reproduces the explicitly given matrix $\mathbf{A}(t)$ in case when latter is of rank at most r for all times $t \in [t_0, T]$ and the initial value $\mathbf{A}(t_0)$ as well as the starting value $\mathbf{Y}(t_0)$ of the integrator coincide. This matrix projector-splitting integrator is an essential fundament for the development of time integration methods for tensors in low-rank formats.

Turning to Tucker tensors, a first time integrator proposed in [Lub15] was launched in the multiconfiguration time-dependent Hartree method [BJWM00] in quantum dynamics and successfully implemented and applied in the field of quantum chemistry [KBL17]. In the quantum physics literature, integration methods have been developed to solve the corresponding equations of motion, see, e.g., [BM97, Lub04], but the appearing inverses of the typically ill-conditioned density matrices lead to severe stepsize restrictions for these numerical integrators. Translating this into the numerical analysis language, the density matrices conform to the matrix $\mathbf{S}(t)$, and we are back in the first proposed matrix integrator [KL07] that suffers from small singular values. In [Lub15], a numerical integrator was presented, which does not deteriorate when small singular values of matricizations of the core tensor $C(t)$ are present. There, first the factor matrices $\mathbf{U}_i(t)$ are propagated in time, for all $i = 1, \ldots, d$, by a repeated application of the first two integration steps of the matrix projector-splitting integrator of [LO14]. The fact that the matrix projector-splitting integrator can be applied to the Tucker tensor case is due to the possibility of matricizing a tensor. At the end of the integration scheme the core tensor $C(t)$ is updated. Though this method propagates each factor of the Tucker decomposition, it does not suffer from small singular values, since it relates back to the favorable matrix projector-splitting integrator that performs well in this situation. Therefore, applying this Tucker integrator to the two-dimensional case yields the same results as the matrix projector-splitting integrator in [LO14]. This integrator is derived in an algorithmic way, but it is also shown in [Lub15] that it can be interpreted as a projector-splitting integrator, decomposing the orthogonal projection $\mathrm{P}(Y(t))$ into a sum of subprojections.

For tensor trains, a time integrator for dynamical tensor approximation in this format was presented in [LOV15]. This method uses a Lie–Trotter splitting of the vector field $\mathrm{P}(Y(t))F(t, Y(t))$, and generates subproblems that are solved sequentially by taking the solution of one substep as the initial value for the next. It turns out that solving the arising subproblems means updating the core tensors $C_i(t)$ of the tensor train representation sequentially. A closer look on the differential equations that need to be solved reveals that this integration method in fact is an extension of the matrix integrator, since matrix differential equations of the same form as in the two-dimensional case are solved here. Again, applying the integration scheme for matrices to the tensor train format is rendered possible due unfolding a tensor into a matrix. Another analogy to the matrix case is the exactness property. Numerical experiments illustrate that this integration scheme has no computational restrictions in the presence of small singular values.

We refer to the survey article [GKT13], which assembles further low-rank formats for

tensors and techniques to approximate them.

This thesis focuses on the above mentioned low-rank formats and follows the ansatz of the dynamical low-rank approximation in order to determine low-rank approximations to explicitly or implicitly given matrices and tensors.

How does this dissertation embed into the landscape of dynamical low-rank approximation of matrices and tensors, which is shaped by the above methods? This thesis contributes to the development of new algorithms, their numerical analysis and the analysis of the already existing numerical integrators. It is build up on the joint work [KLW16] of the author with E. Kieri and Ch. Lubich, on the joint work [OPW18] of the author with A. Ostermann and C. Piazzola as well as on the joint work [LVW18] of the author with Ch. Lubich and B. Vandereycken. Let us explicate the results separately:

Indeed, the exactness property of the matrix integrator was shown in [LO14], though due to the simple but technical nature of the proof, it barely reveals an explanation about this behavior. In this thesis, we give a new proof of the exactness of the matrix projector-splitting integrator that argues with the appearing subprojections and therefore aims to give a deeper insight about this seldom property of a numerical integrator.

The main contribution within the two-dimensional case is the error analysis of the matrix projector-splitting integrator of [LO14], which shows that this integrator is insensitive to the presence of small singular values in the approximation matrix $\mathbf{Y}(t)$. This property is not shared by any standard integrator, such as Runge–Kutta methods, whose behavior deteriorates when singular values become small. We show that if the function F maps onto the tangent bundle of the manifold $\mathcal{M}$ up to an ε-perturbation, the error of the matrix projector-splitting integrator will be bounded in terms of ε and the time step size of the integration procedure. Hence, the closer $F(t, \cdot)$ is to the prescribed rank, i.e., the closer it is to the low-rank manifold, the better approximation results are obtained. The main challenge is to avoid using the Lipschitz constant of the orthogonal projection $\mathrm{P}(Y(t))$, since it behaves proportional to the inverse of the smallest singular value of $\mathbf{Y}(t)$. What comes to our help is the exactness property of the matrix integrator and the preservation of the spaces spanned by the columns of one of the orthonormal matrices $\mathbf{U}(t)$, $\mathbf{V}(t)$, or those of both, respectively, within the integration steps. Combining those two essential properties, the error estimate is obtained with a completely new technique of proof.

The robustness proof of the matrix projector-splitting integrator does not use the local Lipschitz constant of the orthogonal projection, but it requires Lipschitz continuity of the function $F(t, \cdot)$. This gives rise to the application of the result to stiff differential equations, such as discretized partial differential equations, only under a severe CFL condition. However, such a restriction is not observed to be necessary in numerical experiments. We introduce the *low-rank Lie–Trotter splitting integrator*, a numerical method for the low-rank approximation of stiff matrix differential equations, where the right-hand side consists of a stiff, linear and a non-stiff, nonlinear part. Although the right-hand side of the differential equation is stiff, the new integration method has no restrictive choice of the time step size. We also present an error bound, that proves the method to be robust with respect to small singular values *and* which is independent of the possibly large Lipschitz constant of the

full stiff right-hand side of the differential equation. The main idea is to split away the stiff from the non-stiff part and integrate each of the arising subproblems separately. We observe that the stiff part can be solved exactly using semigroup theory [Paz83, EN99] and this avoids using its Lipschitz constant in the error analysis of the proposed method. We also show how the low-rank Lie–Trotter splitting integrator can be applied to the differential Lyapunov and to the differential Riccati equation, two essential representatives of this class of stiff matrix differential equations.

Moreover, we derive a new numerical integrator for the dynamical low-rank approximation of tensors in Tucker format. It differs from the method proposed in [Lub15] in that it follows a conceptually different derivation. The main idea is to apply the matrix projector-splitting integrator to matricizations of the Tucker approximation tensor, where the first two steps are solved exactly and the third step is solved inexactly by a low-rank approximation of the matricized approximation tensor in the subsequent mode. There, we again apply the matrix integrator with inexact solution in the last substep, and so forth. The nested structure of the scheme is eponymous for the *nested Tucker integrator*. The integrator updates the factor matrices $\mathbf{U}_i(t)$ subsequently for each mode $i = 1, \dots, d$ and after each such update, it simply does not take this already updated matrices into account for the next step, i.e., it gets rid of this computational ballast when solving the initial value problem for the intermediate approximation tensor, which is then reduced in dimension. The advantage over the method proposed in [Lub15] is the reduction of the computational complexity in each substep, since the integrator therein solves differential equations for the full (intermediate) approximation tensor in each substep.

As the nested Tucker integrator is mainly based on the matrix projector-splitting integrator, it allows us to transfer the known favorable properties of the matrix integrator to the tensor case. We give a rigorous error analysis of the nested Tucker integrator and prove its robustness in the presence of small singular values of matricizations of the approximation tensor.

Also, we show that in case when the explicitly given tensor $A(t)$ is already of prescribed low rank, the nested Tucker integrator yields the exact solution in the time grid points.

We further show that the newly derived integrator is mathematically equivalent to the Tucker tensor projector-splitting integrator of [Lub15]. Therefore, we show implicitly that this integrator is also exact and its error bound is independent of singular values.

We also present a way to prove the exactness property of the integrator of [Lub15] directly.

Our contribution to the dynamical low-rank approximation of tensor trains is an error analysis of the time integrator introduced in [LOV15]. The error bound given in [KLW16, Theorem 3.1] is robust with respect to small singular values, where an inductive argument that takes the first two steps from the matrix projector-splitting integrator into account is used. Since the integration method for tensor trains is an extension of the matrix integrator, it is comprehensible that an analogous first order error bound can be shown. In this thesis, we will concisely address this result and refer to the full content to the original work [KLW16].

Let us conclude the introduction by giving an outline of this thesis.

The first chapter serves as a basis for the thesis. We recap the (truncated) singular value decomposition, which is the fundamental low-rank representation for matrices throughout this thesis. We also present the idea of the dynamical low-rank approximation, which we study for the two-dimensional case, though it also serves as the central model reduction technique for higher-dimensional tensors. Further, we recall a first integration method for the differential equation for the rank-reduced approximation matrix. This part and the ansatz of the dynamical low-rank approximation are mainly taken from the constitutive paper [KL07]. We discuss difficulties of the discretized dynamical low-rank approximation in the presence of small singular values.

In Chapter 2, we present the projector-splitting integration method proposed in [LO14], which does not suffer from small singular values. We give a new proof for the exactness property of the integrator, which has never been published or submitted before. The main part of this chapter is the proof of robustness of the projector-splitting integrator with respect to small singular values. We also give error bounds for specific situations and exemplify its favorable behavior in this situation within numerical examples. We illustrate that the integrator is applicable to stiff differential equations. The numerical analysis of the projector-splitting integrator is build on [KLW16].

Chapter 3 deals with the question how to compute low-rank approximations to stiff matrix differential equations. We present a numerical integrator based on a Lie–Trotter splitting and perform an error analysis that gives error bounds, which are independent of the Lipschitz constant of the full stiff right-hand side as well as independent of singular values. We apply this method onto two important representatives of the considered class of stiff matrix differential equations: the differential Lyapunov and the differential Riccati equation. All of this chapter is drawn from [OPW18], except the discussion about the low-rank Strang splitting as well as the algorithmic description of the integration procedure.

Chapter 4 is concerned with the time integration of Tucker tensors. First, we amplify the Tucker tensor format as a low-rank representation of tensors. Afterwards, we propose the nested Tucker integrator, a numerical integration method that determines low-rank approximations in Tucker tensor form. We also prove the exactness property of this integrator and perform an error analysis, which shows its robustness regarding small singular values. We further compare the nested Tucker integrator with the Tucker projector-splitting integrator of [Lub15] and prove that they are mathematically equivalent. All this is taken from [LVW18]. We give a direct proof about exactness of the method in [Lub15], which has not been submitted or published elsewhere.

Finally, Chapter 5 consists of two parts. First, we discuss the tensor train format and review the error analysis proposed in [KLW16] of the time integrator of tensor trains presented in [LOV15]. The tensor train decomposition is a special case of the hierarchical Tucker tensor format that has a binary structure and was independently introduced in [HK09] and in [Gra10]. In the second part, we give future prospects about tensor tree networks, a format that is even more general than the hierarchical Tucker tensors. We give a concise introduction into this general tensor format and present an idea of how to integrate tensor tree networks in time.

1 The dynamical low-rank approximation

This introductory chapter is dedicated to set up the environment, which is the base of our considerations for the following chapters. Our aim is to determine an approximation matrix $\mathbf{Y}(t) \in \mathbb{R}^{n_1 \times n_2}$ to a time-dependent matrix $\mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_2}$ under the constraint that the rank r of $\mathbf{Y}(t)$ is much smaller than the rank of $\mathbf{A}(t)$ for all $t_0 \leq t \leq T$, where t denotes a time-variable and $r \in \mathbb{N}$. In classical literature, this low-rank approximation problem is known as finding a best approximation to the matrix $\mathbf{A}(t)$, which is realized by a singular value decomposition truncating rows and columns of the factor matrices. In contrast, we will follow the idea of the *dynamical* low-rank approximation, which requires the time derivative $\dot{\mathbf{Y}}(t)$ of the approximation matrix to be in the tangent space of the approximation manifold such that

$$\|\dot{\mathbf{Y}}(t) - \dot{\mathbf{A}}(t)\| = \min .$$

With $\|\cdot\|$ we denote the Frobenius norm throughout this thesis.

This minimization problem leads to a differential equation for the approximation matrix $\mathbf{Y}(t)$, which needs to be solved numerically. In order to solve this differential equation in an efficient way, we make use of a suitable factorization of low-rank matrices. Here, we choose the decomposition

$$\mathbf{Y}(t) \approx \mathbf{U}(t)\mathbf{S}(t)\,\mathbf{V}(t)^\top, \qquad \text{for all } t_0 \leq t \leq T,$$

as a low-rank representation of $\mathbf{Y}(t)$ and evolve evolution equations for the factor matrices $\mathbf{U}(t) \in \mathbb{R}^{n_1 \times r}$, $\mathbf{S}(t) \in \mathbb{R}^{r \times r}$ and $\mathbf{V}(t) \in \mathbb{R}^{n_2 \times r}$, where $\mathbf{U}(t)$ and $\mathbf{V}(t)$ have orthonormal columns and $r \in \mathbb{N}$ denotes the rank of $\mathbf{Y}(t)$.

We start with recalling the singular value decomposition, which is a fundamental matrix decomposition throughout this thesis. We also give a historic overview of the discovery of the singular value decomposition and its applications, such as the best low-rank approximation of a matrix, see Section 1.1. Afterwards, starting from Section 1.2, we mainly trace ideas and results of [KL07], which is the cornerstone of the dynamical low-rank approximation. In Section 1.2, we first present the ansatz of the dynamical low-rank approximation, leading to a differential equation for the approximation matrix $\mathbf{Y}(t)$. Second, we follow the approach in [KL07] for determining differential equations for the factor matrices out of the differential equation for $\mathbf{Y}(t)$ and integrate them in time, see Section 1.3. Finally, we

discuss this integration method with regard to the presence of small singular values in the approximation matrix in Section 1.4. Here, we focus on the computational point of view, on the behavior of the time-dependent approximation method under consideration concerning a geometric aspect of the approximation manifold and we study the error bound of the integrator. Further approximation properties of this integration method were studied in [KL07].

1.1 The singular value decomposition

We start with a basic concept of matrix decomposition, which we will use in a slightly modified form throughout this thesis. The theory of matrix decompositions can be dated to 1809, where C.F. Gauss made a note about how to reduce binary quadratic forms [Gau09], which he amplified in 1823 in [Gau23] by giving an elimination algorithm. This was the advent of matrix factorizations.

About fifty years later, E. Beltrami in 1873, and C. Jordan in 1874 both considered independently from each other bilinear forms and how to simplify them in order to find their minima and maxima, see [Bel73] and [Jor74], respectively. They detected that for their computations, it is beneficial to first factorize the bilinear form into three matrices, where two of them are required to be orthogonal and the third matrix is assumed to be diagonal. Also, they discovered how to determine each of those three factor matrices. Although they never wrote down the product of those three matrices in the way we denote it nowadays, Beltrami and Jordan are certainly the founders of the existence of the singular value decomposition. Thanks to their pionieering work, we know that each matrix $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2}$ admits a *singular value decomposition* (SVD)

$$\mathbf{A} = \mathbf{W}_1 \, \mathbf{\Sigma} \, \mathbf{W}_2^\top,$$

where $\mathbf{W}_1 \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{W}_2 \in \mathbb{R}^{n_2 \times n_2}$ are orthonormal matrices and

$$\mathbf{\Sigma} = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_{\min\{n_1,n_2\}}) \ \in \ \mathbb{R}^{n_1 \times n_2}$$

is a rectangular diagonal matrix assembling the non-negative, real singular values σ_i for all $i = 1, \ldots, \min\{n_1, n_2\}$.

In fact, Beltrami and Jordan derived the singular value decomposition for a real, square and nonsingular matrix having distinct singular values. For a classical existence proof of the singular value decomposition, we refer to [GVL96].

1.1.1 The SVD as a best approximation

In 1907, E. Schmidt first considered the singular value decomposition (though not denoting it as such) for linear operators and their approximation: in [Sch07, Viertes Kapitel], he worked on integral equations in infinite dimensional spaces and there, he formulated the problem of finding a sum of at most r terms consisting of the product of two functions depending on different variables in order to approximate a continuous function: “Es werde

verlangt [die Funktion] durch eine Summe von höchstens [r] Produkten einer stetigen Funktion [...] mit einer [anderen] stetigen Funktion [...] möglichst gut zu approximieren", see [Sch07, p. 467]. Translating his continuous into a discrete setting, we find that his method is the precursor of the singular value decomposition, where the problem for one variable is formulated as follows: find an approximation to the time-dependent matrix $\mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_2}$ for each $t \in [t_0, T]$ of the form

$$\mathbf{A}(t) \approx \sum_{i=1}^{r} x_i(t) z_i(t)^\top$$

with the constraint

$$\Big\| \mathbf{A}(t) - \sum_{i=1}^{r} x_i(t) z_i(t)^\top \Big\|_{\mathrm{HS}} = \min,$$

where $\|\cdot\|_{\mathrm{HS}}$ is the Hilbert-Schmidt norm. It turns out that when writing the matrix $\mathbf{A}(t)$ in factorized form

$$\mathbf{A}(t) = \mathbf{W}_1(t)\, \mathbf{\Sigma}(t)\, \mathbf{W}_2(t)^\top,$$

with $\mathbf{W}_1(t) \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{W}_2(t) \in \mathbb{R}^{n_2 \times n_2}$ having orthogonal columns and $\mathbf{\Sigma}(t) \in \mathbb{R}^{n_1 \times n_2}$ being diagonal and then truncating columns $(r+1), \ldots, n_1$ of the matrix $\mathbf{W}_1(t)$, as well as columns $(r+1), \ldots, n_2$ of the matrix $\mathbf{W}_2(t)$ and both, rows $(r+1), \ldots, n_1$ and columns $(r+1), \ldots, n_2$ of the matrix $\mathbf{\Sigma}(t)$, such that we obtain matrices of smaller size, i.e.,

$$\begin{aligned}
\mathbf{W}_1(t) \in \mathbb{R}^{n_1 \times n_1} &\xrightarrow{\text{trunc(col}(r+1,\ldots,n_1))} \mathbf{U}(t) \in \mathbb{R}^{n_1 \times r}, \\
\mathbf{W}_2(t) \in \mathbb{R}^{n_2 \times n_2} &\xrightarrow{\text{trunc(col}(r+1,\ldots,n_2))} \mathbf{V}(t) \in \mathbb{R}^{n_2 \times r}, \\
\mathbf{\Sigma}(t) \in \mathbb{R}^{n_1 \times n_2} &\xrightarrow{\text{trunc(row}(r+1,\ldots,n_1)),\ \text{trunc(col}(r+1,\ldots,n_2))} \mathbf{S}(t) \in \mathbb{R}^{r \times r},
\end{aligned}$$

the resulting matrix

$$\mathbf{Y}(t) := \sum_{i=1}^{r} \sigma_i(t) u_i(t) v_i(t)^\top = \mathbf{U}(t) \mathbf{S}(t)\, \mathbf{V}(t)^\top,$$

where $u_i(t)$ and $v_i(t)$ are the columns of $\mathbf{U}(t)$ and of $\mathbf{V}(t)$, respectively, and with decreasing $\sigma_i(t)$ for increasing $i = 1, \ldots, r$, is the *best approximation* to $\mathbf{A}(t)$ for each $t \in [t_0, T]$ with respect to the condition that $\operatorname{rank} \mathbf{Y}(t) \leq r$:

$$\min \Big\| \mathbf{A}(t) - \sum_{i=1}^{r} x_i(t) z_i(t)^\top \Big\|_{\mathrm{HS}} = \big\| \mathbf{A}(t) - \mathbf{Y}(t) \big\|_{\mathrm{HS}}.$$

In this way, Schmidt turned the SVD from a theoretical result to a mathematical tool, which nowadays is indispensable for matrix approximations. In [Sch07, §19], Schmidt gives a proof of his theorem for this best approximation, which he formulates in words as "Das [...] definierte Maß der besten Approximation [...] einer Funktion [...] durch eine

Summe von höchstens [r] Produkten einer Funktion [...] mit einer [anderen] Funktion [...] verschwindet bei [...] wachsendem [r]".

In 1936, C. Eckart and G. Young have determined the best approximation of one matrix to another by defining a distance function, which can be obtained by manipulating its terms, see [EY36]. They also show that the truncated SVD is the best approximation of a matrix by another matrix of low rank.

L. Mirsky joins this group of developers of the best approximation of a matrix. He extended this result to unitary invariant norms, such as the Frobenius norm and the spectral norm for matrices in 1960. He used the same techniques in his proof as Schmidt, but his motivation was different, since he came up with this minimization problem by studying properties of unitary invariant matrix norms, see [Mir60]. He arrived at the same conclusion as Schmidt, Eckart and Young and therefore this result is often called the Schmidt-Eckart-Young-Mirsky Theorem in mathematical literature. We study how to obtain this result in the spectral norm denoted by $\|\cdot\|_2$, where we omit the time dependence:

Without loss of generality, let $n_1 \geq n_2$ and let $\mathbf{A} = \sum_{i=1}^{n_2} \sigma_i u_i v_i^\top$ be given in the singular value decomposed form, where σ_i are the singular values of $\mathbf{A}$ and u_i and v_i denote the orthonormal columns of the matrices $\mathbf{W}_1$ and $\mathbf{W}_2$, respectively. Let $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2}$ with

$$\mathbf{Y} = \sum_{i=1}^{r} \sigma_i u_i v_i^\top,$$

then we have

$$\mathbf{A} - \mathbf{Y} = \sum_{i=r+1}^{n_2} \sigma_i u_i v_i^\top$$

and σ_{r+1} is the largest singular value of $\mathbf{A} - \mathbf{Y}$. Therefore, by definition of the spectral norm we find

$$\|\mathbf{A} - \mathbf{Y}\|_2 = \sigma_{r+1}.$$

Let further $\mathbf{Z} \in \mathbb{R}^{n_1 \times n_2}$ with $\operatorname{rank} \mathbf{Z} = r$ and $r \ll n_2$. Then it follows by the rank-nullity theorem that

$$\operatorname{rank} \mathbf{Z} + \ker \mathbf{Z} = n_2 \quad \Longleftrightarrow \quad \ker \mathbf{Z} = n_2 - r.$$

Then,

$$\dim(\ker \mathbf{Z}) + \dim(\operatorname{span}\{v_1, \ldots, v_{r+1}\}) = (n_2 - r) + (r + 1) = n_2 + 1 \geq n_2.$$

Hence, $\ker \mathbf{Z} \cap \operatorname{span}\{v_1, \ldots, v_{r+1}\} \neq \emptyset$. For $z \in \ker \mathbf{Z} \cap \operatorname{span}\{v_1, \ldots, v_{r+1}\}$ with $\|z\|_2 = 1$, we have $\mathbf{Z}\, z = (0, \ldots, 0)^\top$ and $z = \alpha_1 v_1 + \cdots + \alpha_{r+1} v_{r+1}$, for $\alpha_i \in \mathbb{R} \setminus \{0\}$ for all $i = 1, \ldots, r + 1$. Therefore, with

$$\|z\|_2^2 = \sum_{i=1}^{r+1} |\alpha_i|^2 = 1,$$

we have

$$\begin{aligned}
\|\mathbf{A}-\mathbf{Z}\|_2^2 &= \|\mathbf{A}-\mathbf{Z}\|_2^2\ \|z\|_2^2 \\
&\geq \|(\mathbf{A}-\mathbf{Z})z\|_2^2 = \|\mathbf{A}\,z\|_2^2 = \left\|\sum_{i=1}^{r+1}\alpha_i\,\mathbf{A}\,v_i\right\|_2^2 \\
&= \left\|\sum_{i=1}^{r+1}\alpha_i\sigma_i u_i v_i^\top v_i\right\|_2^2 = \sum_{i=1}^{r+1}|\alpha_i\sigma_i|^2 \\
&\geq \sum_{i=1}^{r+1}|\alpha_i\sigma_{r+1}|^2 = \sigma_{r+1}^2\sum_{i=1}^{r+1}|\alpha_i|^2 = \sigma_{r+1}^2 \\
&= \|\mathbf{A}-\mathbf{Y}\|_2^2,
\end{aligned}$$

which shows that $\mathbf{Y}$ is the best approximation matrix of rank r to the matrix $\mathbf{A}$.

1.1.2 Reduction of computational cost

Schmidts contribution in determining the best approximation matrix of lower rank to a given matrix with full rank opened the way for the singular value decomposition being a crucial computational tool. Eckart and Young give the first approach of how to find the approximation matrix $\mathbf{Y}$ and they are the first who name this procedure a "truncated singular value decomposition".

When assuming $r \ll \{n_1, n_2\}$, we can save computational memory and time: instead of storing the full matrix $\mathbf{A} \in \mathbb{R}^{n_1\times n_2}$, which means saving $(n_1 \cdot n_2)$ entries, we only store the factor matrices of the truncated singular value decomposition $\mathbf{Y} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top$. Due to the dimensions of the factor matrices, this requires the storage of only $(n_1 r + r^2 + n_2 r)$ entries.

Hence, in favor of computational efficiency and for reducing the complexity of the problems we will deal with throughout this thesis, we consider the rank r to be significantly smaller than the dimensions of the matrix.

1.2 Ansatz of the dynamical low-rank approximation

Formulated for the two-dimensional case, the overall problem throughout this thesis is the following: let $\mathbf{A}(t) \in \mathbb{R}^{n_1\times n_2}$, find an approximation matrix

$$\mathbf{Y}(t) \in \mathbb{R}^{n_1\times n_2}, \quad \text{with} \quad \operatorname{rank}\mathbf{Y}(t) = r,$$

to $\mathbf{A}(t)$ for all $t_0 \leq t \leq T$, where $r \ll \{n_1, n_2\}$. A *best approximation* matrix $\mathbf{Y}(t)$, which, compared to the matrix $\mathbf{A}(t)$, is of low rank r and hence is in the low-rank manifold $\mathcal{M}$ satisfies

$$\mathbf{Y}(t) \in \mathcal{M} \qquad \text{such that} \qquad \|\mathbf{Y}(t) - \mathbf{A}(t)\| = \min. \tag{1.1}$$

By the Schmidt-Eckart-Young-Mirsky Theorem discussed in Section 1.1, we know that a truncated singular value decomposition solves this minimization problem.

Although the truncated SVD results in the best approximation to the given problem, it has several drawbacks. From the computational point of view, it is expensive to compute a truncated SVD at each time t with regard to computational time and memory. Also, computing a pointwise best approximation does not necessarily yield a smooth approximation matrix-valued function $t \mapsto \mathbf{Y}(t)$. Further, the SVD does not result in a unique decomposition: with orthogonal matrices $\mathbf{Q}_1 \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{Q}_2 \in \mathbb{R}^{n_2 \times n_2}$, we can replace $\widetilde{\mathbf{U}} = \mathbf{U}\,\mathbf{Q}_1$, $\widetilde{\mathbf{V}} = \mathbf{V}\,\mathbf{Q}_2$ and $\widetilde{\mathbf{S}} = \mathbf{Q}_1^\top\,\mathbf{S}\,\mathbf{Q}_2$ and then we have $\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top = \mathbf{Y} = \widetilde{\mathbf{U}}\widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^\top \in \mathcal{M}$.

A method, which does not suffer from those disadvantages, i.e., is computationally feasible, results in a smooth approximation $\mathbf{Y}(t)$ and works on unique representations of the factor matrices, follows the ansatz of the *dynamical low-rank approximation* given in [KL07]. There, instead of the best approximation as above, a *low-rank approximation* $\mathbf{Y}(t) \in \mathcal{M}$ is determined from the condition that for every $t \in [t_0, T]$, the time derivative $\dot{\mathbf{Y}}(t)$, which is in the tangent space $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$, satisfies

$$\dot{\mathbf{Y}}(t) \in \mathcal{T}_{\mathbf{Y}(t)}\mathcal{M} \qquad \text{such that} \qquad \|\dot{\mathbf{Y}}(t) - \dot{\mathbf{A}}(t)\| = \min. \tag{1.2}$$

The ansatz of computing the best approximation (1.1) and the approach (1.2) for computing a low-rank approximation both are reasonable for the case when the matrix $\mathbf{A}(t)$ that needs to be approximated is given explicitly. This is different in case when $\mathbf{A}(t)$ is known implicitly: an essential benefit of the approach (1.2) is its extension to the case, when the matrix $\mathbf{A}(t)$ is given as the unknown solution to a matrix differential equation

$$\dot{\mathbf{A}}(t) = F(t, \mathbf{A}(t)), \qquad \mathbf{A}(t_0) = \mathbf{A}^0. \tag{1.3}$$

Here, we replace $\dot{\mathbf{A}}(t)$ in (1.2) by $F(t, \mathbf{Y}(t))$, such that the minimization problem for the implicit case reads

$$\dot{\mathbf{Y}}(t) \in \mathcal{T}_{\mathbf{Y}(t)}\mathcal{M} \qquad \text{such that} \qquad \|\dot{\mathbf{Y}}(t) - F(t, \mathbf{Y}(t))\| = \min. \tag{1.4}$$

Hence, the approximation matrix $\mathbf{Y}(t)$ is determined from the condition, that its time derivative, which is in the tangent space $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$ is chosen, such that the residual in the differential equation is minimized. In other words, out of all elements $\delta\,\mathbf{Y}$ in the tangent space $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$, the matrix $\dot{\mathbf{Y}}(t)$ is the one that minimizes the defect. Therefore, condition (1.4) is equivalent to the task of finding $\dot{\mathbf{Y}}(t) \in \mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$ that satisfies

$$\langle \dot{\mathbf{Y}}(t) - F(t, \mathbf{Y}(t)), \delta\,\mathbf{Y} \rangle = 0 \qquad \text{for all} \quad \delta\,\mathbf{Y} \in \mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}. \tag{1.5}$$

From the numerical analysis perspective, this is a Galerkin condition on the tangent space $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$. In fact, it was P.A.M. Dirac from the quantum mechanics community, who used this orthogonality condition (1.5) in 1930 in order to find an approximation to the time-dependent Schrödinger equation, see [Dir30a, Dir30b]. Since condition (1.5) holds for all tangent matrices $\delta\,\mathbf{Y}$, they can vary, which is why this ansatz is also known as variational principle in the quantum mechanics literature.

A few years later, J.I. Frenkel interpreted Dirac's orthogonality condition (1.5) as the minimization condition (1.4), see [Fre34]. This shows that $\dot{\mathbf{Y}}(t)$ is the orthogonal projection of $F(t, \mathbf{Y}(t))$ onto $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$. Denoting the orthogonal projection operator onto the tangent

space of the low-rank manifold by $\mathrm{P}(\mathbf{Y}(t))$, we have the differential equation on $\mathcal{M}$ given by

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y}(t))F(t, \mathbf{Y}(t)), \tag{1.6}$$

which is equivalent to condition (1.5) as well as to the minimization problem (1.4). In the fundamental work of O. Koch and Ch. Lubich [KL07], this condition is proposed to be the ansatz of the *dynamical low-rank approximation*. In the quantum dynamics literature it is well known as the *Dirac–Frenkel variational principle*, see, e.g., [Lub08, II.1]. Based on this principle, there are several results in the chemical physics literature amongst others, which propose computationally efficient methods for determining approximations in quantum molecular dynamics, e.g., the multiconfiguration time-dependent Hartree method [BJWM00]. A near-optimality result for variational approximations based on the Dirac–Frenkel variational principle is given in [Lub05].

Thinking in terms of the low-rank manifold $\mathcal{M}$ and the orthogonal projection onto the tangent space $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$, we imagine condition (1.6) as

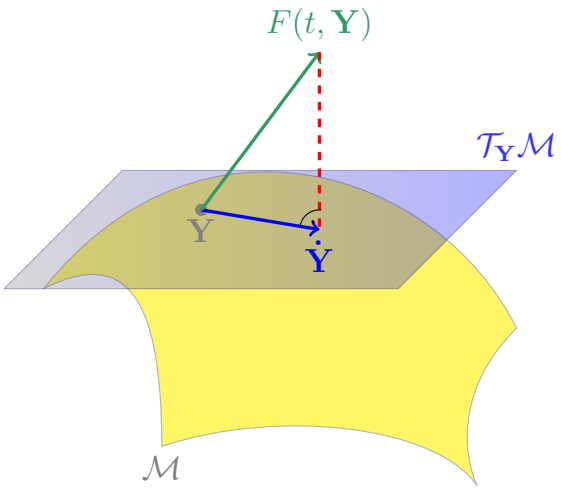

Figure 1.1: Orthogonal projection onto the tangent space of the low-rank manifold. The red dashed line represents the orthogonal projection, which results in $\dot{\mathbf{Y}}$. Out of all $\delta\,\mathbf{Y} \in \mathcal{T}_{\mathbf{Y}}\mathcal{M}$, $\dot{\mathbf{Y}}$ is the tangent element that minimizes the distance between $F(t, \mathbf{Y})$ and the tangent space of $\mathcal{M}$ at the approximation matrix $\mathbf{Y}$.

Providing an initial value $\mathbf{Y}(t_0) = \mathbf{Y}^0 \in \mathcal{M}$ of low rank for the differential equation (1.6), the resulting initial value problem

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y}(t))F(t, \mathbf{Y}(t)), \qquad \mathbf{Y}(t_0) = \mathbf{Y}^0 \tag{1.7}$$

needs to be solved numerically in order to obtain a low-rank approximation $\mathbf{Y}(t)$ to the matrix $\mathbf{A}(t)$ given in (1.3), with $\mathbf{Y}(t_0) \approx \mathbf{A}(t_0)$. Determining the approximation matrix

$\mathbf{Y}(t)$ requires solving the ordinary differential equation (ODE) (1.7) and hence contrary to the best approximation, the ansatz of the dynamical low-rank approximation yields a smooth approximation matrix $\mathbf{Y}(t)$.

Although the approximation matrix $\mathbf{Y}(t)$ is of low rank, it is still of size $n_1 \times n_2$. Hence, if the size of $\mathbf{Y}(t)$ is large, then solving (1.7) becomes expensive from the computational perspective. In order to determine the solution of (1.7) in an efficient way, the idea is to profit from the fact that $\mathbf{Y}(t)$ is of rank r by choosing a rank r decomposition of the approximation matrix, which is obtained by the SVD. Note that here we do not need the truncated SVD, since $\mathbf{Y}(t)$ is already of low rank r. We study this approach in the subsequent section.

We remark that if the matrix $\mathbf{A}(t_0)$ is given explicitly, we project $\dot{\mathbf{A}}(t)$ orthogonally onto the tangent space $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$, such that the resulting initial value problem is given by

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y}(t))\dot{\mathbf{A}}(t), \qquad \mathbf{Y}(t_0) = \mathbf{Y}^0 .$$

Compared to the best approximation, solving the above initial value problem requires only the increments $\dot{\mathbf{A}}(t)$ instead of the matrix $\mathbf{A}(t)$. In situations where the time derivative of $\mathbf{A}(t)$ is sparser than the matrix itself, we can save computational time and memory when determining the approximation matrix $\mathbf{Y}(t)$ via the dynamical low-rank approximation.

Throughout this work, we will focus on the case when $\mathbf{A}(t)$ is given as the unknown solution of the differential equation (1.3). Our results also hold for the explicit case by simply choosing $F(t, \mathbf{Y}(t))$ as $\dot{\mathbf{A}}(t)$. If we change this perspective, we emphasize it in the particular situations, e.g., in the exactness results in Section 2.2.1 as well as in Section 4.4 for Tucker tensors and in Section 5.1 for tensor trains.

1.3 An integration method

The first approach for solving the differential equation (1.7) for the approximation matrix evolved in the previous section is proposed in [KL07]. In this section, we follow the idea to use a SVD-like decomposition of $\mathbf{Y}(t)$ and derive differential equations for its factors. Then, after having solved the initial value problems for the factors $\mathbf{U}(t)$, $\mathbf{S}(t)$ and $\mathbf{V}(t)$, we obtain the desired approximation matrix $\mathbf{Y}(t)$.

1.3.1 Unique representation of the tangent factor matrices

The differential equation (1.7) that we have to solve in order to obtain a low-rank approximation $\mathbf{Y}(t)$ is equivalent to finding its time derivative $\dot{\mathbf{Y}}(t) \in \mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$ such that the defect of the given differential equation (1.3) is orthogonal to $\mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$ *for all* tangent elements $\delta\,\mathbf{Y}$ in the tangent space of the low-rank manifold $\mathcal{M}$. But which form does $\delta\,\mathbf{Y}$ admit? We first take a look on their representation in the tangent space at $\mathbf{Y}(t) \in \mathcal{M}$.

Moreover, contrary to the SVD, which is not a unique factorization of $\mathbf{Y}(t)$ in general, see Section 1.1, we require a unique decomposition of the tangent matrix $\delta\,\mathbf{Y} \in \mathcal{T}_{\mathbf{Y}(t)}\mathcal{M}$ as well as of the tangent matrices for the factors $\mathbf{U}$, $\mathbf{S}$ and $\mathbf{V}$ (omitting the time dependence).

From Section 1.1 we know that each rank r matrix $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2}$ admits a SVD factorization. Here, we do not require the full-rank matrix $\mathbf{S}$ to be diagonal and hence we consider the SVD-like decomposition

$$\begin{aligned} &\mathbf{Y} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top, \\ \text{with} \quad &\text{orthonormal } \mathbf{U} \in \mathbb{R}^{n_1 \times r}, \text{i.e.,} \quad \mathbf{U}^\top \mathbf{U} = \mathbf{I}_{n_1}, \\ &\text{orthonormal } \mathbf{V} \in \mathbb{R}^{n_2 \times r}, \text{ i.e.,} \quad \mathbf{V}^\top \mathbf{V} = \mathbf{I}_{n_2} \\ \text{and} \quad &\text{(non-)diagonal } \mathbf{S} \in \mathbb{R}^{r \times r}. \end{aligned}$$

Let us collect the orthonormal matrices $\mathbf{U}$ and $\mathbf{V}$ in the manifolds

$$\mathcal{V}_{n_1,r} := \{\mathbf{U} \in \mathbb{R}^{n_1 \times r} | \, \mathbf{U}^\top \mathbf{U} = \mathbf{I}_{n_1}\} \quad \text{and} \quad \mathcal{V}_{n_2,r} := \{\mathbf{V} \in \mathbb{R}^{n_2 \times r} | \, \mathbf{V}^\top \mathbf{V} = \mathbf{I}_{n_2}\},$$

respectively, where each of them is a Stiefel manifold [Sti35, §1]. Now, since the SVD exists for every matrix $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2}$, there exists a map

$$\begin{aligned} \mathbb{R}^{r \times r} \times \mathcal{V}_{n_1,r} \times \mathcal{V}_{n_2,r} \;&\to\; \mathbb{R}^{n_1 \times n_2}, \\ (\mathbf{S}, \mathbf{U}, \mathbf{V}) \;&\mapsto\; \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top = \mathbf{Y}, \end{aligned}$$

which is surjective as the SVD in general is a non-unique decomposition. In order to assure uniqueness in the tangent space, we extend this map to become a bijection for the tangent elements. To this end, we denote by

$$\begin{aligned} &\mathcal{T}_{\mathbf{U}}\mathcal{V}_{n_1,r} := \{\delta\,\mathbf{U} \in \mathbb{R}^{n_1 \times r} | \delta\,\mathbf{U}^\top \mathbf{U} + \mathbf{U}^\top \delta\,\mathbf{U} = \mathbf{0}\} = \{\delta\,\mathbf{U} \in \mathbb{R}^{n_1 \times r} | \, \mathbf{U}^\top \delta\,\mathbf{U} \in \mathrm{skew}_r\}, \\ \text{and} \quad &\mathcal{T}_{\mathbf{V}}\mathcal{V}_{n_2,r} := \{\delta\,\mathbf{V} \in \mathbb{R}^{n_2 \times r} | \delta\,\mathbf{V}^\top \mathbf{V} + \mathbf{V}^\top \delta\,\mathbf{V} = \mathbf{0}\} = \{\delta\,\mathbf{V} \in \mathbb{R}^{n_2 \times r} | \, \mathbf{V}^\top \delta\,\mathbf{V} \in \mathrm{skew}_r\}, \end{aligned}$$

the tangent spaces to the Stiefel manifolds $\mathcal{V}_{n_1,r}$ and $\mathcal{V}_{n_2,r}$ at the orthonormal matrices $\mathbf{U}$ and $\mathbf{V}$, respectively. With those tangent spaces at hand, we consider the linear map

$$\begin{aligned} \mathbb{R}^{r \times r} \times \mathcal{T}_{\mathbf{U}}\mathcal{V}_{n_1,r} \times \mathcal{T}_{\mathbf{V}}\mathcal{V}_{n_2,r} \;&\to\; \mathcal{T}_{\mathbf{Y}}\mathcal{M} \times \mathrm{skew}_r \times \mathrm{skew}_r, \\ (\delta\mathbf{S}, \delta\,\mathbf{U}, \delta\,\mathbf{V}) \;&\mapsto\; (\delta\,\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\delta\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\delta\,\mathbf{V}^\top = \delta\,\mathbf{Y},\ \mathbf{U}^\top \delta\,\mathbf{U},\ \mathbf{V}^\top \delta\,\mathbf{V}). \end{aligned}$$

Counting the dimensions of the domain and of the codomain, respectively, we see that they coincide and hence we conclude that this map is surjective. Moreover, the kernel of this linear map is $(\mathbf{0}, \mathbf{0}, \mathbf{0})$, which is why it is also injective and therefore in fact, this map is an isomorphism. This means that each element in the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$ of the low-rank manifold at the approximation matrix $\mathbf{Y}$ admits the representation

$$\delta\,\mathbf{Y} = \delta\,\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\delta\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\delta\,\mathbf{V}^\top. \tag{1.8}$$

Further, since $\mathbf{U}^\top \delta\,\mathbf{U}$ and $\mathbf{V}^\top \delta\,\mathbf{V}$ are skew-symmetric matrices, we impose the orthogonality constraints

$$\mathbf{U}^\top \delta\,\mathbf{U} = \mathbf{0} \qquad \text{and} \qquad \mathbf{V}^\top \delta\,\mathbf{V} = \mathbf{0} \tag{1.9}$$

in order to obtain unique representations of the tangent matrices for $\mathbf{S}$, $\mathbf{U}$ and $\mathbf{V}$: multiplying the form of $\delta\,\mathbf{Y}$ given in (1.8) by $\mathbf{U}^\top$ from the left and by $\mathbf{V}$ from the right, we find

$$\delta\mathbf{S} = \mathbf{U}^\top\,\delta\,\mathbf{Y}\,\mathbf{V}\,. \tag{1.10}$$

Further, since the matrix $\mathbf{S} \in \mathbb{R}^{r\times r}$ is of full rank, its inverse exists. Therefore, multiplying $\delta\,\mathbf{Y}$ first by $\mathbf{V}$, then by $\mathbf{S}^{-1}$, both, from the right, and using the form (1.10) of $\delta\mathbf{S}$ yields

$$\delta\,\mathbf{U} = (\mathbf{I}_{n_1} - \mathbf{U}\,\mathbf{U}^\top)\delta\,\mathbf{Y}\,\mathbf{V}\,\mathbf{S}^{-1}. \tag{1.11}$$

Similarly, we obtain a unique representation of $\delta\,\mathbf{V}$ by multiplying (1.8) from the left by $\mathbf{U}^\top$ and by $\mathbf{S}^{-1}$ such that

$$\delta\,\mathbf{V} = (\mathbf{I}_{n_2} - \mathbf{V}\,\mathbf{V}^\top)\delta\,\mathbf{Y}^\top\,\mathbf{U}\,\mathbf{S}^{-\top}. \tag{1.12}$$

1.3.2 Differential equations for the factor matrices

The underlying representation of the low-rank matrix that is sought admits the form $\mathbf{Y}(t) = \mathbf{U}(t)\mathbf{S}(t)\,\mathbf{V}(t)^\top$. Using this decomposition, the time derivative of the low-rank matrix $\mathbf{Y}(t)$ is determined by the Leibniz rule, which results in

$$\dot{\mathbf{Y}}(t) = \dot{\mathbf{U}}(t)\mathbf{S}(t)\,\mathbf{V}(t)^\top + \mathbf{U}(t)\dot{\mathbf{S}}(t)\,\mathbf{V}(t)^\top + \mathbf{U}(t)\mathbf{S}(t)\dot{\mathbf{V}}(t)^\top. \tag{1.13}$$

Moreover, since $\mathbf{U}(t)$ and $\mathbf{V}(t)$ have orthonormal columns, we have $\mathbf{U}(t)^\top\,\mathbf{U}(t) = \mathbf{I}_{n_1}$ and $\mathbf{V}(t)^\top\,\mathbf{V}(t) = \mathbf{I}_{n_2}$, respectively. Again, by the Leibniz rule, we obtain

$$\dot{\mathbf{U}}(t)^\top\,\mathbf{U}(t) + \mathbf{U}(t)^\top\dot{\mathbf{U}}(t) = \mathbf{0}_{n_1} \qquad \text{and} \qquad \dot{\mathbf{V}}(t)^\top\,\mathbf{V}(t) + \mathbf{V}(t)^\top\dot{\mathbf{V}}(t) = \mathbf{0}_{n_2}$$

and hence, translating the orthogonality conditions (1.9) to this time-dependent setting, we require

$$\mathbf{U}(t)^\top\dot{\mathbf{U}}(t) = \mathbf{0}_{n_1} \qquad \text{and} \qquad \mathbf{V}(t)^\top\dot{\mathbf{V}}(t) = \mathbf{0}_{n_2}. \tag{1.14}$$

In the following, we will use representation (1.13) for the time derivative of $\mathbf{Y}$, but for ease of presentation we omit using the time dependence t.

With the unique representations (1.10)-(1.12) of the tangent factor matrices and the unique form (1.8) of $\delta\,\mathbf{Y}$ at hand, we are now in the situation to deduce differential equations for the factor matrices $\mathbf{S}$, $\mathbf{U}$ and $\mathbf{V}$.

Recalling that the minimization condition (1.4) is equivalent to the Galerkin condition (1.5), viz.,

$$\langle \dot{\mathbf{Y}} - F(t,\mathbf{Y}), \delta\,\mathbf{Y}\rangle = 0 \qquad \text{for all} \quad \delta\,\mathbf{Y} \in \mathcal{T}_{\mathbf{Y}}\mathcal{M},$$

the idea is to choose $\delta\,\mathbf{Y}$ in a way that provides ordinary differential equations for the factor matrices. Since the Galerkin condition holds for all tangent matrices

$$\delta\,\mathbf{Y} = \delta\,\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\delta\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\delta\,\mathbf{V}^\top,$$

we first choose $\delta\mathbf{U} = \delta\mathbf{V} = \mathbf{0}$. With the form (1.13) of $\dot{\mathbf{Y}}$ and with the orthogonality constraints (1.14), this yields

$$\begin{aligned}0 &= \langle \dot{\mathbf{Y}} - \dot{\mathbf{A}}, \mathbf{U}\,\delta\mathbf{S}\,\mathbf{V}^\top \rangle \\ &= \langle \mathbf{U}^\top \dot{\mathbf{U}}\mathbf{S}\,\mathbf{V}^\top\mathbf{V} + \mathbf{U}^\top\mathbf{U}\dot{\mathbf{S}}\,\mathbf{V}^\top\mathbf{V} + \mathbf{U}^\top\mathbf{U}\,\delta\mathbf{S}\dot{\mathbf{V}}^\top\mathbf{V} - \mathbf{U}^\top\dot{\mathbf{A}}\,\mathbf{V}, \delta\mathbf{S}\rangle \\ &= \langle \dot{\mathbf{S}} - \mathbf{U}^\top\dot{\mathbf{A}}\,\mathbf{V}, \delta\mathbf{S}\rangle.\end{aligned}$$

It follows that

$$\dot{\mathbf{S}} = \mathbf{U}^\top\dot{\mathbf{A}}\,\mathbf{V}\,. \tag{1.15}$$

Next, for deriving the differential equation for $\mathbf{U}$, we choose $\delta\mathbf{Y} \in \mathcal{T}_{\mathbf{Y}}\mathcal{M}$ to be of the form $\delta\mathbf{Y} = \delta\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top$, i.e., $\delta\mathbf{S} = \delta\mathbf{V} = \mathbf{0}$. Then, by using the form (1.15) of $\dot{\mathbf{S}}$, we find

$$\begin{aligned}0 &= \langle \dot{\mathbf{Y}} - \dot{\mathbf{A}}, \delta\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top \rangle \\ &= \langle \dot{\mathbf{U}}\mathbf{S} + \mathbf{U}\dot{\mathbf{S}} - \dot{\mathbf{A}}\,\mathbf{V}, \delta\mathbf{U}\,\mathbf{S}\rangle \\ &= \langle \dot{\mathbf{U}}\mathbf{S}\mathbf{S}^\top + \mathbf{U}\dot{\mathbf{S}}\mathbf{S}^\top - \dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top, \delta\mathbf{U}\rangle \\ &= \langle \dot{\mathbf{U}}\mathbf{S}\mathbf{S}^\top + \mathbf{U}\,\mathbf{U}^\top\dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top - \dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top, \delta\mathbf{U}\rangle.\end{aligned}$$

Now, due to the orthogonality condition $\mathbf{U}^\top\delta\mathbf{U} = \mathbf{0}_{n_1}$, the tangent matrix lies in the range of the orthogonal complement of the space spanned by the columns of $\mathbf{U}$. In other words, by defining

$$\mathrm{P}_{\mathbf{U}} = \mathbf{U}\,\mathbf{U}^\top \qquad \text{and} \qquad \mathrm{P}_{\mathbf{U}}^\perp = \mathbf{I}_{n_1} - \mathbf{U}\,\mathbf{U}^\top,$$

as the orthogonal projections onto the spaces spanned by the columns of $\mathbf{U}$ as well as the orthogonal projection onto the complements of those spaces, we conclude that $\delta\mathbf{U} \in$ range $\mathrm{P}_{\mathbf{U}}^\perp$. Therefore, there exists an arbitrary $\delta\mathbf{W} \in \mathbb{R}^{n_1\times n_2}$, such that $\delta\mathbf{U} = \mathrm{P}_{\mathbf{U}}^\perp\delta\mathbf{W}$. Hence, it follows that

$$\begin{aligned}0 &= \langle \dot{\mathbf{U}}\mathbf{S}\mathbf{S}^\top + \mathbf{U}\,\mathbf{U}^\top\dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top - \dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top, \mathrm{P}_{\mathbf{U}}^\perp\delta\mathbf{W}\rangle \\ &= \langle \mathrm{P}_{\mathbf{U}}^\perp\dot{\mathbf{U}}\mathbf{S}\mathbf{S}^\top + \mathrm{P}_{\mathbf{U}}^\perp\mathbf{U}\,\mathbf{U}^\top\dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top - \mathrm{P}_{\mathbf{U}}^\perp\dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top, \delta\mathbf{W}\rangle \\ &= \langle \dot{\mathbf{U}}\mathbf{S}\mathbf{S}^\top - \mathrm{P}_{\mathbf{U}}^\perp\dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top, \delta\mathbf{W}\rangle.\end{aligned}$$

Since this holds for an arbitrary matrix $\delta\mathbf{W} \in \mathbb{R}^{n_1\times n_2}$, it yields

$$\mathbf{0} = \dot{\mathbf{U}}\mathbf{S}\mathbf{S}^\top - \mathrm{P}_{\mathbf{U}}^\perp\dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^\top$$

and by multiplying by $\mathbf{S}^{-\top}$ as well as by $\mathbf{S}^{-1}$ from the right, we obtain a differential equation for the factor matrix $\mathbf{U}$, which is of the form

$$\dot{\mathbf{U}} = \mathrm{P}_{\mathbf{U}}^\perp\dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{S}^{-1}. \tag{1.16}$$

The derivation of the differential equation for the factor matrix $\mathbf{V}$ goes along similar lines. The Galerkin condition (1.5) holds for all tangent matrices $\delta\mathbf{Y} \in \mathcal{T}_{\mathbf{Y}}\mathcal{M}$ and in particular for $\delta\mathbf{Y} = \mathbf{U}\,\mathbf{S}\delta\mathbf{V}^\top$, i.e., when choosing $\delta\mathbf{U} = \delta\mathbf{S} = \mathbf{0}$. Inserting this $\delta\mathbf{Y}$ into

(1.5), using the representation (1.13) of the time derivative of $\mathbf{Y}$, adhering the orthogonality constraints (1.14) and inserting the time derivative (1.15) of $\mathbf{S}$ results in

$$\begin{aligned}
0 &= \langle \dot{\mathbf{Y}} - \dot{\mathbf{A}}, \mathbf{U}\,\mathbf{S}\delta\,\mathbf{V}^\top \rangle \\
&= \langle \mathbf{U}^\top \dot{\mathbf{U}}\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}^\top \mathbf{U}\,\dot{\mathbf{S}}\,\mathbf{V}^\top + \mathbf{U}^\top \mathbf{U}\,\mathbf{S}\dot{\mathbf{V}}^\top - \mathbf{U}^\top \dot{\mathbf{A}}, \mathbf{S}\delta\,\mathbf{V}^\top \rangle \\
&= \langle \mathbf{S}^\top \dot{\mathbf{S}}\,\mathbf{V}^\top + \mathbf{S}^\top \mathbf{S}\dot{\mathbf{V}}^\top - \mathbf{S}^\top \mathbf{U}^\top \dot{\mathbf{A}}, \delta\,\mathbf{V}^\top \rangle \\
&= \langle \mathbf{S}^\top \mathbf{U}^\top \dot{\mathbf{A}}\,\mathbf{V}\,\mathbf{V}^\top + \mathbf{S}^\top \mathbf{S}\dot{\mathbf{V}}^\top - \mathbf{S}^\top \mathbf{U}^\top \dot{\mathbf{A}}, \delta\,\mathbf{V}^\top \rangle \\
&= \langle \mathbf{V}\,\mathbf{V}^\top \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S} + \dot{\mathbf{V}}\mathbf{S}^\top \mathbf{S} - \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S}, \delta\,\mathbf{V} \rangle.
\end{aligned}$$

Defining

$$\mathrm{P}_{\mathbf{V}} = \mathbf{V}\,\mathbf{V}^\top \qquad \text{and} \qquad \mathrm{P}_{\mathbf{V}}^{\perp} = \mathbf{I}_{n_2} - \mathbf{V}\,\mathbf{V}^\top$$

to be the orthogonal projection onto the space spanned by the columns of $\mathbf{V}$ and the orthogonal projection onto the complement space, we conclude by the orthogonality condition $\mathbf{V}^\top \delta\,\mathbf{V} = \mathbf{0}$ in (1.9) that $\delta\,\mathbf{V} \in \text{range } \mathrm{P}_{\mathbf{V}}^{\perp}$. Hence, there exists an arbitrary $\delta\,\mathbf{Z} \in \mathbb{R}^{n_1 \times n_2}$, such that $\delta\,\mathbf{V} = \mathrm{P}_{\mathbf{V}}^{\perp}\delta\,\mathbf{Z}$. Therefore, we find

$$\begin{aligned}
0 &= \langle \mathbf{V}\,\mathbf{V}^\top \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S} + \dot{\mathbf{V}}\mathbf{S}^\top \mathbf{S} - \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S}, \delta\,\mathbf{V} \rangle \\
&= \langle \dot{\mathbf{V}}\mathbf{S}^\top \mathbf{S} - \mathrm{P}_{\mathbf{V}}^{\perp} \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S}, \mathrm{P}_{\mathbf{V}}^{\perp}\delta\,\mathbf{Z} \rangle \\
&= \langle \mathrm{P}_{\mathbf{V}}^{\perp}\dot{\mathbf{V}}\mathbf{S}^\top \mathbf{S} - \mathrm{P}_{\mathbf{V}}^{\perp} \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S}, \delta\,\mathbf{Z} \rangle \\
&= \langle \dot{\mathbf{V}}\mathbf{S}^\top \mathbf{S} - \mathrm{P}_{\mathbf{V}}^{\perp} \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S}, \delta\,\mathbf{Z} \rangle.
\end{aligned}$$

This equation holds for any $\delta\,\mathbf{Z} \in \mathbb{R}^{n_1 \times n_2}$ and so we conclude that

$$\mathbf{0} = \dot{\mathbf{V}}\mathbf{S}^\top \mathbf{S} - \mathrm{P}_{\mathbf{V}}^{\perp} \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S},$$

which by multiplying first by $\mathbf{S}^{-1}$ and second by $\mathbf{S}^{-\top}$ from the right yields a differential equation for $\mathbf{V}$, which then is of the form

$$\dot{\mathbf{V}} = \mathrm{P}_{\mathbf{V}}^{\perp} \dot{\mathbf{A}}^\top \mathbf{U}\,\mathbf{S}^{-\top}. \tag{1.17}$$

The idea of the integration method proposed in [KL07] is to solve the system of differential equations (1.15)-(1.17) for the factor matrices of the low-rank representation of $\mathbf{Y}$ with appropriately given initial values $\mathbf{U}(t_0)$, $\mathbf{S}(t_0)$ and $\mathbf{V}(t_0)$, which ideally come from a truncated SVD of the initial value $\mathbf{A}(t_0)$ of the differential equation (1.3). In a nutshell, we solve the system of equations

$$\begin{aligned}
\dot{\mathbf{S}}(t) &= \mathbf{U}(t)^\top \dot{\mathbf{A}}(t)\,\mathbf{V}(t), & \mathbf{S}(t_0) &= \mathbf{S}^0, \\
\dot{\mathbf{U}}(t) &= \mathrm{P}_{\mathbf{U}}^{\perp} \dot{\mathbf{A}}(t)\,\mathbf{V}(t)\mathbf{S}(t)^{-1}, & \mathbf{U}(t_0) &= \mathbf{U}^0, \\
\dot{\mathbf{V}}(t) &= \mathrm{P}_{\mathbf{V}}^{\perp} \dot{\mathbf{A}}(t)^\top \mathbf{U}(t)\mathbf{S}(t)^{-\top}, & \mathbf{V}(t_0) &= \mathbf{V}^0,
\end{aligned} \tag{1.18}$$

with initial values $\mathbf{S}(t_0) \in \mathbb{R}^{r \times r}$, $\mathbf{U}(t_0) \in \mathbb{R}^{n_1 \times r}$ and $\mathbf{V}(t_0) \in \mathbb{R}^{n_2 \times r}$, respectively. Solving (1.18) for one time step $t_0 \to t_0 + h = t_1$ with time step size $h > 0$ and multiplying the

resulting factor matrices by each other yields the low-rank approximation matrix $\mathbf{Y}(t)$ to $\mathbf{A}(t)$ at time $t = t_1$, i.e.,

$$\mathbf{Y}(t_1) = \mathbf{U}(t_1)\mathbf{S}(t_1)\,\mathbf{V}(t_1)^\top.$$

Note that in practice, in order to avoid expensive computations, we actually never build this product.

To continue with the integration procedure in time, we simply take the factor matrices at time t_1 as initial values for the next time integration from $t_1 \to t_1 + h = t_2$ and so forth.

We remark that it is important to assure orthonormality of the initial values for the factor matrices $\mathbf{U}(t)$ and $\mathbf{V}(t)$ at subsequent time steps. In other words, the solutions of the differential equations for $\mathbf{U}(t)$ and $\mathbf{V}(t)$ must stay orthonormal. Now, due to the orthogonality conditions (1.14), the time derivatives of $\mathbf{U}(t)^\top\,\mathbf{U}(t)$ and of $\mathbf{V}(t)^\top\,\mathbf{V}(t)$ are given by

$$\dot{\mathbf{U}}(t)^\top\,\mathbf{U}(t) + \mathbf{U}(t)^\top\dot{\mathbf{U}}(t) = \mathbf{0}_{n_1} \qquad \text{and} \qquad \dot{\mathbf{V}}(t)^\top\,\mathbf{V}(t) + \mathbf{V}(t)^\top\dot{\mathbf{V}}(t) = \mathbf{0}_{n_2},$$

respectively. This means that $\mathbf{U}(t)^\top\,\mathbf{U}(t)$ and $\mathbf{V}(t)^\top\,\mathbf{V}(t)$ are constant and since the initial values $\mathbf{U}(t_0)$ and $\mathbf{V}(t_0)$ are supposed to have orthonormal columns, we conclude that $\mathbf{U}(t)$ and $\mathbf{V}(t)$ retain orthonormal columns during the time-integration of the system (1.18). From the computational perspective, when integrating this system of ODEs numerically, this property is preserved by, e.g., orthogonality-preserving Runge–Kutta methods described in [HLW06, IV.9].

1.3.3 Schematic illustration of the integrator

Throughout this thesis, we will present several numerical integrators. For ease of understanding their scheme, we visualize their structure and the procedure within figures. To this end, we depict vectors, matrices and tensors by nodes having one, two and more branches in the following way:

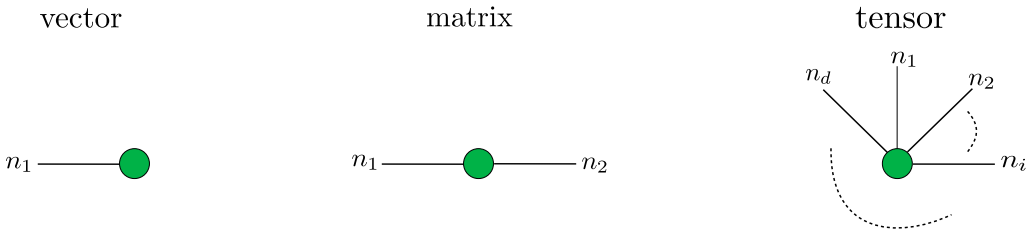

Figure 1.2: Schematic illustration of one-, two- and high-dimensional objects.

Note that the number of branches illustrates the dimension of the object. Hence, we imagine to increase the dimension of an object by sticking more and more branches to the node.

By means of this illustration, we depict a matrix-matrix multiplication, such as the product $\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top = \mathbf{Y}$ for $\mathbf{U} \in \mathbb{R}^{n_1 \times r}$, $\mathbf{V} \in \mathbb{R}^{n_2 \times r}$ and $\mathbf{S} \in \mathbb{R}^{r \times r}$ as

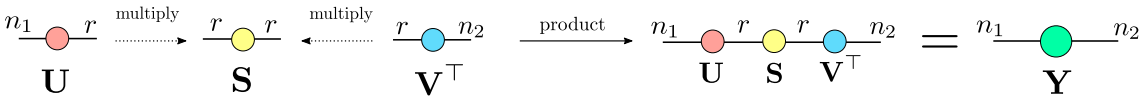

Figure 1.3: Schematic illustration of a matrix-matrix product.

In terms of those figures, we think of a matrix-matrix multiplication as merging branches with the same dimension, such that they fuse to one branch. The tensor-matrix product will be explained and depicted in Figure 4.1 in Chapter 4, where we deal with Tucker tensors.

With those figures at hand, we are now in the situation to illustrate the structure of the integration method described in Section 1.3.2. Suppose that an initial value $\mathbf{Y}^0$, which approximates $\mathbf{A}^0$ is given. Then, following the integration method, we first perform a SVD of $\mathbf{Y}^0$ in order to obtain initial values for the differential equations for the factor matrices and afterwards we solve the system of ODEs (1.18), which yields updated factor matrices $\mathbf{U}^1$, $\mathbf{S}^1$ and $\mathbf{V}^1$ after one time step. Finally, this results in an approximation matrix $\mathbf{Y}^1 \approx \mathbf{A}(t_1)$. The procedure is depicted in the following figure:

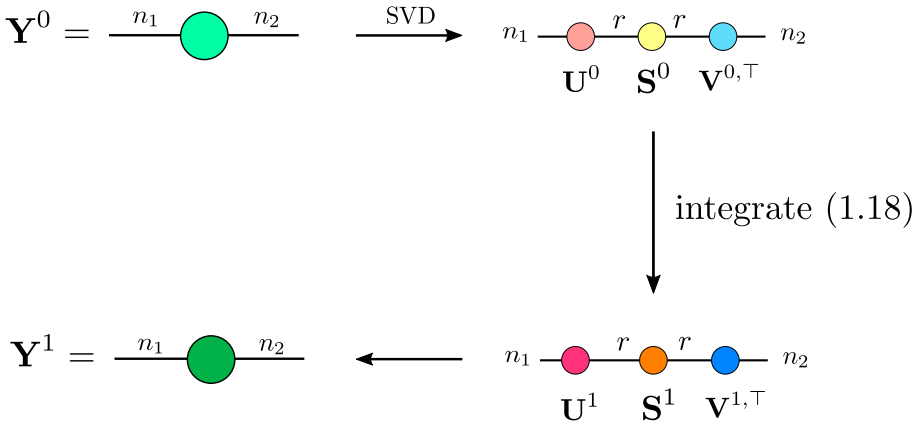

Figure 1.4: Schematic illustration of one time step of the integration method described in Section 1.3.2.

1.4 Discussion about the discretized dynamical low-rank approximation in the presence of small singular values

The presence of small singular values in the low-rank approximation of a large matrix is very common. Unless the matrix has a distinct gap in the distribution of its singular values, truncating all the smallest singular values below a tolerance ε yields a remaining matrix of reduced rank that still has singular values of magnitude $O(\varepsilon)$. Even if there is a distinct gap in the singular value distribution such that two groups of large and negligibly small singular values, respectively, are formed, it is typically not known a priori at which rank

the former group ends. We are also in a time-dependent setting where the distribution of singular values may change over time. Underestimating the effective rank means we neglect a significant part of the matrix, which leads to poor accuracy, but overestimating the effective rank yields an approximation with small singular values.

1.4.1 Computational aspect

A first difficulty with small singular values can be seen in the system of differential equations (1.18). The non-zero singular values of $\mathbf{Y}(t)$ are those of the $r \times r$ matrix $\mathbf{S}(t)$, whose inverse appears in the last two differential equations for $\mathbf{U}(t)$ and for $\mathbf{V}(t)$, respectively. The presence of small singular values therefore leads to severe problems when these differential equations are integrated numerically by standard methods such as Runge–Kutta methods.

1.4.2 Curvature of the low-rank manifold

Another difficulty with small singular values is related to the curvature of the low-rank manifold $\mathcal{M}$. In [KL07, Lemma 4.2], the authors compare the orthogonal projection of an arbitrary matrix $\mathbf{B} \in \mathbb{R}^{n_1 \times n_2}$ at $\mathbf{Y} \in \mathcal{M}$ and at $\widetilde{\mathbf{Y}} \in \mathcal{M}$, where the smallest non-zero singular value $\sigma_r(\widetilde{\mathbf{Y}})$ of $\widetilde{\mathbf{Y}}$ is bounded from below by

$$\sigma_r(\widetilde{\mathbf{Y}}) \geq \rho > 0.$$

Further, they suppose that the distance of the two matrices $\mathbf{Y}$ and $\widetilde{\mathbf{Y}}$ at which the orthogonal projection is available, is bounded by

$$\|\mathbf{Y} - \widetilde{\mathbf{Y}}\| \leq \frac{1}{8}\rho.$$

Then, the difference of the orthogonal projection of the matrix $\mathbf{B}$ at $\mathbf{Y}$ and at $\widetilde{\mathbf{Y}}$, respectively, is bounded by

$$\|\mathrm{P}(\mathbf{Y})\,\mathbf{B} - \mathrm{P}(\widetilde{\mathbf{Y}})\,\mathbf{B}\| = \|(\mathrm{P}(\mathbf{Y}) - \mathrm{P}(\widetilde{\mathbf{Y}}))\,\mathbf{B}\| \leq 8\rho^{-1}\|\mathbf{Y} - \widetilde{\mathbf{Y}}\|\|\mathbf{B}\|_2. \tag{1.19}$$

The proof of this bound is given in [KL07, Proof of Lemma 4.2]. Studying the right-hand side of this estimate, we observe that it depends on the inverse of the smallest non-zero singular value of $\widetilde{\mathbf{Y}}$. Translating this result to our differential equation (1.7), viz.,

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y})F(t, \mathbf{Y}), \qquad \mathbf{Y}(t_0) = \mathbf{Y}^0,$$

this means that the local Lipschitz constant of the tangent space projection P at $\mathbf{Y}$, is proportional to the inverse of the smallest non-zero singular value of $\mathbf{Y}$. Now, in case when the approximation matrix $\mathbf{Y}$ has only large singular values, this is a valuable result, since then ρ^{-1} becomes small. In contrast, if $\mathbf{Y}$ has small singular values, then the local Lipschitz constant of the orthogonal projection $\mathrm{P}(\mathbf{Y})$ becomes large und so the local Lipschitz constant of the full right-hand side of the differential equation is unbearably large for an error analysis of the integration method described in Section 1.3, where this Lipschitz constant is used. This observation has an impact on the error analysis of the integration

method presented in Chapter 2. Illustrating the above bound (1.19), we see the relation between this observation and the curvature of the low-rank manifold:

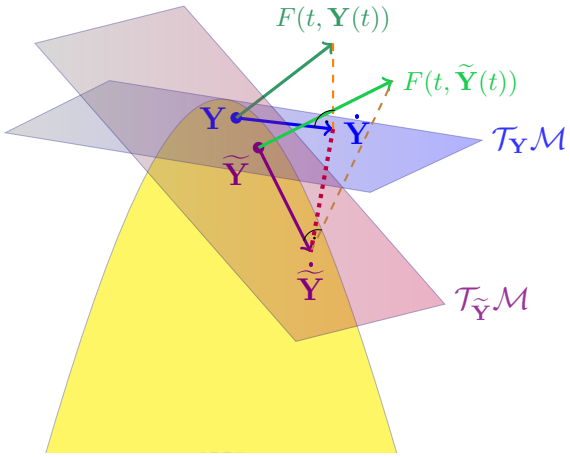

Figure 1.5: Effect of the orthogonal projection onto the low-rank manifold at matrices with small singular values. We see that the higher the curvature, the larger the distance between the tangent spaces onto which we project orthogonally.

The local Lipschitz constant of the tangent space projection $\mathrm{P}(\mathbf{Y})$ is a measure of the curvature of $\mathcal{M}$ at $\mathbf{Y}$. We thus observe that the curvature of the low-rank manifold $\mathcal{M}$ corresponds to the size of the singular values: it is proportional to the inverse of the smallest non-zero singular value of $\mathbf{Y}$. From the numerical point of view, we conclude with the arguments in Section 1.4.1 that the integration method described in Section 1.3 does not perform well in regions on $\mathcal{M}$, where the curvature is high.

1.4.3 Error bound

Following the dynamical low-rank approximation approach described in Section 1.2 and solving the differential equations for the factor matrices of the low-rank representation as presented in Section 1.3, produces an approximation error, which we will discuss in this section.

In [KL07, Theorem 6.1], the authors show local quasi-optimality of this method, where they compare the dynamical low-rank approximation $\mathbf{Y}(t)$ with the best approximation $\mathbf{X}(t)$, in case when a continuously best approximation $\mathbf{X}(t) \in \mathcal{M}$ to the solution $\mathbf{A}(t)$ of the differential equation (1.3) exists for all $t_0 \leq t \leq T$. In the situation when

- the r-th singular value of the best approximation $\mathbf{X}(t)$ has a lower bound, i.e.,

$$\sigma_r(\mathbf{X}(t)) \geq \rho > 0$$

- the best approximation $\mathbf{X}(t)$ is bounded by

$$\|\mathbf{X}(t) - \mathbf{A}(t)\| \leq \frac{1}{16}\rho \tag{1.20}$$

- F is bounded by $B > 0$ along the approximations $\mathbf{X}(t)$ and $\mathbf{Y}(t)$, i.e.,

$$\|F(t, \mathbf{X}(t))\| \leq B, \qquad \|F(t, \mathbf{Y}(t))\| \leq B$$

- F satisfies a one-sided Lipschitz condition, i.e.,

$$\exists\ \lambda \in \mathbb{R},\ \text{such that} \quad \langle F(t, \mathbf{Y}) - F(t, \widetilde{\mathbf{Y}}), \mathbf{Y} - \widetilde{\mathbf{Y}} \rangle \leq \lambda \|\mathbf{Y} - \widetilde{\mathbf{Y}}\|^2, \quad \text{for all } \mathbf{Y}, \widetilde{\mathbf{Y}} \in \mathcal{M}$$

- for the best approximation $\mathbf{X}(t)$ it holds that

$$\exists\ L > 0,\ \text{such that} \quad \|F(t, \mathbf{X}(t)) - F(t, \mathbf{A}(t))\| \leq L \|\mathbf{X}(t) - \mathbf{A}(t)\|,$$

for all $t_0 \leq t \leq T$, the approximation error with initial value $\mathbf{Y}(t_0) = \mathbf{X}(t_0)$ is bounded by

$$\begin{aligned} \|\mathbf{Y}(t) - \mathbf{A}(t)\| &\leq \|\mathbf{Y}(t) - \mathbf{X}(t)\| + \|\mathbf{X}(t) - \mathbf{A}(t)\| \\ &\leq \left(\frac{1}{16} B \rho^{-1} + L \right) \mathrm{e}^{(16 B \rho^{-1} + \lambda) t} \int_{t_0}^{t} \|\mathbf{X}(s) - \mathbf{A}(s)\| \, \mathrm{d}s + \frac{1}{16} \rho. \end{aligned} \tag{1.21}$$

For a proof of this result we refer to [KL07, Theorem 6.1].

In the case when the lower bound ρ of the smallest singular value of the best approximation is reasonably large, this error estimate is suitable due to the inverse of ρ on the right-hand side of the estimate. On the other hand, in case when the lower bound of $\sigma_r(\mathbf{X}(t))$ is small, its inverse on the right-hand side of (1.21) leads to a large contribution of the exponential to the error bound of the low-rank approximation. Hence, this error bound for the integration method described in Section 1.3, which is given in terms of the best approximation $\mathbf{X}(t)$ to the full-rank solution $\mathbf{A}(t)$, is not significant, if the smallest singular value of that best approximation is undersized. Then, the error $\|\mathbf{Y}(t) - \mathbf{A}(t)\|$ might be small, but this is not reflected in the error bound (1.21).

Moreover, the situation that the best approximation is $(1/16)\rho$-distance away from the full-rank solution $\mathbf{A}(t)$ as assumed in (1.20) does not mirror a general setting, but is rather specific. In case when the bound ρ of the smallest singular value $\sigma_r(\mathbf{X}(t))$ is small, it means that $\mathbf{A}(t)$ is close to its best approximation $\mathbf{X}(t)$. Therefore, we conclude that either $\mathbf{X}(t)$ would be (almost) of full rank or $\mathbf{A}(t)$ is (almost) of low rank, i.e., it is close to the low-rank manifold $\mathcal{M}$ in regions where its curvature is high, see Figure 1.5.

Furthermore, in the explicit situation, when there is no distinct gap in the distribution of the singular values and when

$$\mathbf{A}(t) = \mathbf{M}(t) + \mathbf{R}(t),$$

where $\mathbf{M}(t) \in \mathcal{M}$ and $\|\dot{\mathbf{R}}(t)\| \leq \varepsilon$, an $\mathcal{O}(\varepsilon)$ error is proven, even for (small) singular values $\rho \sim \varepsilon$ in [KL07, Theorem 5.5.], which can be extended to the implicit case.

2 Error analysis of the matrix projector-splitting integrator

The objective of this chapter is to show that the matrix projector-splitting integrator is insensitive to the presence of small singular values, a property that is not shared by any standard integrator such as explicit or implicit Runge–Kutta methods, whose behavior deteriorate when singular values become small.

The presence of small singular values in the low-rank approximation of a large matrix is very common. In case when the matrix has small singular values and we choose the approximation rank r to be larger than the effective rank, then we are overestimating the matrix and this yields an approximation matrix with small singular values.

Here, we are concerned with time-dependent matrices $\mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_2}$, $t_0 \leq t \leq T$, for large n_1 and n_2. These matrices are either known explicitly, or are the unknown solution of a matrix differential equation

$$\dot{\mathbf{A}}(t) = F(t, \mathbf{A}(t)), \qquad \mathbf{A}(t_0) = \mathbf{A}^0 \,. \tag{2.1}$$

We seek an approximate solution $\mathbf{Y}(t)$ to (2.1) on the manifold $\mathcal{M}$ of rank $r \ll \{n_1, n_2\}$ matrices of size $n_1 \times n_2$. To construct an evolution equation for $\mathbf{Y}(t) \in \mathcal{M}$ we follow the concept of projecting the right-hand side of the differential equation orthogonally onto the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$ of $\mathcal{M}$ at the current approximation matrix $\mathbf{Y}(t)$, just as described in Chapter 1. This yields the differential equation for $\mathbf{Y}(t)$ on the manifold $\mathcal{M}$,

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y}(t))F(t, \mathbf{Y}(t)), \qquad \mathbf{Y}(t_0) = \mathbf{Y}^0 \in \mathcal{M}. \tag{2.2}$$

In order to solve this differential equation in an efficient way, we follow the matrix projector-splitting integrator proposed by Ch. Lubich and I.V. Oseledets [LO14], which we recall in Section 2.1, by first deriving the integration scheme and then giving a practical integration procedure. Afterwards, in Section 2.2, we will discuss two essential properties of the matrix projector-splitting integrator: first, we will deal with the exactness property of the integrator, which was proven in [LO14, Theorem 4.1], but we give a new proof in Section 2.2.1, which uses the appearing subprojections within the integration scheme and therefore aims to give a deeper insight about the exactness. This proof was neither published nor submitted elsewhere before. Second, we will recall in Section 2.2.2 the property that the matrix projector-splitting integrator preserves either the range or the corange or both of the current approximation matrix, see [LO14, Lemma 3.1]. Those two properties enable

us to give error bounds that are independent of small singular values. This error analysis is published in [KLW16] by the author in collaboration with E. Kieri and Ch. Lubich. Here, we give the proof in much more detail and in a partly modified way in Section 2.3. Afterwards, in Section 2.4, we handle three specific situations and give error bounds for those cases, see [KLW16]. All error bounds are measured in the Frobenius nom $\|\cdot\|$. Finally, we will corroborate our theoretical results by several numerical examples in Section 2.5, where we mainly follow the experiments in [KLW16], but additionally discuss error bounds of the Strang splitting, see [Str68], in the presence of small singular values.

In the following, we will sometimes refer to the matrix projector-splitting integrator simply as projector-splitting integrator or matrix integrator.

2.1 The matrix projector-splitting integrator

To enter the stage, we will first present the integration method for determining low-rank approximations to (2.2). This integration scheme is fundamental with regard to integrating higher-dimensional tensors, since their integration follows the same principle, see later Chapter 4.

After having derived the integrator, we will also give the algorithm in Section 2.1.2. We will carry out this section by following the lines of [LO14], where the matrix projector-splitting integrator was proposed. Further details can be read therein, since we will restrict ourselves on presenting the integration method.

2.1.1 Deriving the integrator

In this section, we will derive the projector-splitting integrator, which was proposed in [LO14].

Our aim is to solve equation (2.2) for the approximation matrix $\mathbf{Y}(t)$ of low rank r in an efficient way. Further, we want to improve the method described in Section 1.3 with regard to small singular values.

The approximation matrix $\mathbf{Y}(t)$ is in the low-rank manifold and therefore it can be factorized as $\mathbf{Y}(t) = \mathbf{U}(t)\mathbf{S}(t)\,\mathbf{V}(t)^\top$. From the derivation of the integration method in Section 1.3, where the factor matrices of $\mathbf{Y}(t)$ are integrated, we have an explicit form (1.18) of the derivatives of the factors $\mathbf{U}, \mathbf{S}$ and $\mathbf{V}$. Using this representation yields

$$\begin{aligned}
\dot{\mathbf{Y}} &= \dot{\mathbf{U}}\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\dot{\mathbf{S}}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\dot{\mathbf{V}}^\top \\
&= \big((\mathbf{I} - \mathbf{U}\,\mathbf{U}^\top)F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{S}^{-1}\big)\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\big(\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\big)\,\mathbf{V}^\top \\
&\quad + \mathbf{U}\,\mathbf{S}\big((\mathbf{I} - \mathbf{V}\,\mathbf{V}^\top)F(t,\mathbf{Y})^\top\,\mathbf{U}\,\mathbf{S}^{-\top}\big)^\top \\
&= F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top - \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y}),
\end{aligned}$$

where we have (partially) omitted the time dependence for ease of presentation. Setting this equal to the representation (2.2) of $\dot{\mathbf{Y}}$ in terms of the orthogonal projection $\mathrm{P}(\mathbf{Y})$ onto the tangent space of the low-rank manifold, results in

$$\mathrm{P}(\mathbf{Y})F(t,\mathbf{Y}) = F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top - \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y}). \tag{2.3}$$

We observe that the orthogonal projection $\mathrm{P}(\mathbf{Y})$ can be written as a sum of three subprojections consisting of the orthogonal projectors $\mathbf{U}\,\mathbf{U}^\top$ and $\mathbf{V}\,\mathbf{V}^\top$. Denoting the three summands on the right-hand side in terms of projections as

$$\begin{aligned}\mathrm{P}_1^+ F(t,\mathbf{Y}) &= F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top,\\ \mathrm{P}_1^- F(t,\mathbf{Y}) &= \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top,\\ \mathrm{P}_2^+ F(t,\mathbf{Y}) &= \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y}),\end{aligned}$$

gives us the equivalent expression

$$\mathrm{P}(\mathbf{Y})F(t,\mathbf{Y}) = \mathrm{P}_1^+ F(t,\mathbf{Y}) - \mathrm{P}_1^- F(t,\mathbf{Y}) + \mathrm{P}_2^+ F(t,\mathbf{Y}).$$

The fact that $\mathrm{P}(\mathbf{Y})F(t,\mathbf{Y})$ is in $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$ does not guarantee that each term on the right-hand side of (2.3) is also in this tangent space of the low-rank manifold. However, we observe that projecting the three terms separately onto the tangent space of the current approximation matrix gives

$$\begin{aligned}\mathrm{P}(\mathbf{Y})\big(F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big) &= \big(F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big)\,\mathbf{V}\,\mathbf{V}^\top - \mathbf{U}\,\mathbf{U}^\top\big(F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big)\,\mathbf{V}\,\mathbf{V}^\top\\ &\quad + \mathbf{U}\,\mathbf{U}^\top\big(F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big)\\ &= F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top,\end{aligned}$$

$$\begin{aligned}\mathrm{P}(\mathbf{Y})\big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big) &= \big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big)\,\mathbf{V}\,\mathbf{V}^\top\\ &\quad - \mathbf{U}\,\mathbf{U}^\top\big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big)\,\mathbf{V}\,\mathbf{V}^\top\\ &\quad + \mathbf{U}\,\mathbf{U}^\top\big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top\big)\\ &= \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\,\mathbf{V}\,\mathbf{V}^\top,\end{aligned}$$

$$\begin{aligned}\mathrm{P}(\mathbf{Y})\big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\big) &= \big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\big)\,\mathbf{V}\,\mathbf{V}^\top - \mathbf{U}\,\mathbf{U}^\top\big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\big)\,\mathbf{V}\,\mathbf{V}^\top\\ &\quad + \mathbf{U}\,\mathbf{U}^\top\big(\mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y})\big)\\ &= \mathbf{U}\,\mathbf{U}^\top F(t,\mathbf{Y}),\end{aligned}$$

and so each of the three terms in (2.3) is in the tangent space of the low-rank manifold $\mathcal{M}$. Therefore, each auxiliary matrix $\mathbf{Y}_i^\pm(t) \in \mathcal{M}$, with $i = 1$ and $\pm$ as well as with $i = 2$ and $+$ that satisfies

$$\dot{\mathbf{Y}}_i^\pm(t) = \pm\mathrm{P}_i^\pm(\mathbf{Y})F(t,\mathbf{Y}_i^\pm(t)), \qquad \mathbf{Y}_i^\pm(t_0) = \mathbf{Y}_i^{\pm,0}, \tag{2.4}$$

stays in the low-rank manifold and so it can be factorized as

$$\mathbf{Y}_i^\pm = \mathbf{U}_i^\pm\,\mathbf{S}_i^\pm\,\mathbf{V}_i^{\pm,\top}. \tag{2.5}$$

Our study about the orthogonal projection $\mathrm{P}(\mathbf{Y})$ leads us to two important observations. First, we see that the projection can be rewritten in terms of a composition of three subprojections $\mathrm{P}_i^\pm(\mathbf{Y})$. Second, projecting $F(t,\mathbf{Y})$ by each of those subprojections yields a matrix that is already in the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$ and therefore, solutions $\mathbf{Y}_i^\pm$ of subproblems (2.4) stay in the low-rank manifold. These observations suggest that instead of integrating

$$\dot{\mathbf{Y}}(t) = \mathrm{P}_1^+ F(t,\mathbf{Y}) - \mathrm{P}_1^- F(t,\mathbf{Y}) + \mathrm{P}_2^+ F(t,\mathbf{Y}), \qquad \mathbf{Y}(t_0) = \mathbf{Y}^0 \tag{2.6}$$

directly, we follow the Lie–Trotter splitting method, where we solve (2.4) for $1^+, 1^-$ and 2^+. We formulate the matrix projector-splitting integrator from $t_0 \to t_1$ and for $t_0 \leq t \leq t_1$ as:

1. $\mathbf{Y}_1^+$**-step**: Update $\mathbf{Y}_1^+(t_0) \to \mathbf{Y}_1^+(t_1)$ by solving

$$\dot{\mathbf{Y}}_1^+(t) = \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))F(t, \mathbf{Y}_1^+(t)), \qquad \mathbf{Y}_1^+(t_0) = \mathbf{Y}^0 \tag{2.7}$$

2. $\mathbf{Y}_1^-$**-step**: Update $\mathbf{Y}_1^-(t_0) \to \mathbf{Y}_1^-(t_1)$ by solving

$$\dot{\mathbf{Y}}_1^-(t) = -\mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))F(t, \mathbf{Y}_1^-(t)), \qquad \mathbf{Y}_1^-(t_0) = \mathbf{Y}_1^+(t_1) \tag{2.8}$$

3. $\mathbf{Y}_2^+$**-step**: Update $\mathbf{Y}_2^+(t_0) \to \mathbf{Y}_2^+(t_1)$ by solving

$$\dot{\mathbf{Y}}_2^+(t) = \mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))F(t, \mathbf{Y}_2^+(t)), \qquad \mathbf{Y}_2^+(t_0) = \mathbf{Y}_1^-(t_1). \tag{2.9}$$

Finally, take $\mathbf{Y}^1 = \mathbf{Y}_2^+(t_1) \approx \mathbf{Y}(t_1)$ as an approximation to the solution of (2.2). This is the actual matrix projector-splitting integrator formulated in an abstract way.

It is possible to simplify the above integration scheme by using the fact that each solution of the subproblems in the above integration scheme stays in the low-rank manifold, if the initial value is in there. Hence, with (2.5), a comparison of both sides of (2.7) yields

$$\begin{aligned} &\dot{\mathbf{U}}_1^+\mathbf{S}_1^+\,\mathbf{V}_1^{+,\top} + \mathbf{U}_1^+\,\dot{\mathbf{S}}_1^+\,\mathbf{V}_1^{+,\top} + \mathbf{U}_1^+\,\mathbf{S}_1^+\dot{\mathbf{V}}_1^{+,\top} = F(t, \mathbf{Y}_1^+)\,\mathbf{V}_1^+\,\mathbf{V}_1^{+,\top} \\ \iff \quad &\big(\dot{\mathbf{U}_1^+\mathbf{S}_1^+}\big)\,\mathbf{V}_1^{+,\top} + \mathbf{U}_1^+\,\mathbf{S}_1^+\dot{\mathbf{V}}_1^{+,\top} = F(t, \mathbf{Y}_1^+)\,\mathbf{V}_1^+\,\mathbf{V}_1^{+,\top}. \end{aligned}$$

The last equality is satisfied if

$$\big(\dot{\mathbf{U}_1^+\mathbf{S}_1^+}\big) = F(t, \mathbf{Y}_1^+)\,\mathbf{V}_1^+ \qquad \text{and} \qquad \dot{\mathbf{V}}_1^+ = 0. \tag{2.10}$$

Hence, $\mathbf{V}_1^+ \in \mathbb{R}^{n_2 \times r}$ is a constant orthonormal matrix. For the second subproblem (2.8), we have

$$\dot{\mathbf{U}}_1^-\mathbf{S}_1^-\,\mathbf{V}_1^{-,\top} + \mathbf{U}_1^-\,\dot{\mathbf{S}}_1^-\,\mathbf{V}_1^{-,\top} + \mathbf{U}_1^-\,\mathbf{S}_1^-\dot{\mathbf{V}}_1^{-,\top} = -\,\mathbf{U}_1^-\,\mathbf{U}_1^{-,\top}\,F(t, \mathbf{Y}_1^-)\,\mathbf{V}_1^-\,\mathbf{V}_1^{-,\top},$$

which holds true if

$$\dot{\mathbf{S}}_1^- = -\,\mathbf{U}_1^{-,\top}\,F(t, \mathbf{Y}_1^-)\,\mathbf{V}_1^- \qquad \text{and} \qquad \dot{\mathbf{U}}_1^- = \dot{\mathbf{V}}_1^- = 0 \tag{2.11}$$

and similarly for the third supbroblem, we find

$$\begin{aligned} &\dot{\mathbf{U}}_2^+\mathbf{S}_2^+\,\mathbf{V}_2^{+,\top} + \mathbf{U}_2^+\,\dot{\mathbf{S}}_2^+\,\mathbf{V}_2^{+,\top} + \mathbf{U}_2^+\,\mathbf{S}_2^+\dot{\mathbf{V}}_2^{+,\top} = \mathbf{U}_2^+\,\mathbf{U}_2^{+,\top}\,F(t, \mathbf{Y}_2^+) \\ \iff \quad &\dot{\mathbf{U}}_2^+\mathbf{S}_2^+\,\mathbf{V}_2^{+,\top} + \mathbf{U}_2^+\big(\mathbf{S}_2^+\,\dot{\mathbf{V}}_2^{+,\top}\big) = \mathbf{U}_2^+\,\mathbf{U}_2^{+,\top}\,F(t, \mathbf{Y}_2^+) \\ \iff \quad &\big(\mathbf{S}_2^+\,\dot{\mathbf{V}}_2^{+,\top}\big) = \mathbf{U}_2^{+,\top}\,F(t, \mathbf{Y}_2^+) \qquad \text{and} \qquad \dot{\mathbf{U}}_2{}^+ = 0. \end{aligned} \tag{2.12}$$

We can already observe at this stage that the matrices $\mathbf{V}_1^\pm$ stay constant during the time integration of the first two and the matrices $\mathbf{U}_1^-$, $\mathbf{U}_2^+$ are not changed during the last two integration steps. Therefore, we simply do not consider them in the following

scheme of the projector-splitting integrator. Also, we drop the indices and superscripts for ease of notation and generalize notation by $\mathbf{U}$ and $\mathbf{V}$. We will distinguish these basis matrices by denoting their time dependence in their superscripts. Renaming $\mathbf{U}\,\mathbf{S} =: \mathbf{K}$ and $\mathbf{S}\,\mathbf{V}^\top =: \mathbf{L}^\top$, we conclude that solving the system of differential equations (2.7)–(2.9) within the Lie–Trotter splitting integrator is equivalent to solving

1. **K-step**: Update $\mathbf{U}^0 \to \mathbf{U}^1$, $\mathbf{S}^0 \to \widehat{\mathbf{S}}^1$ by solving

$$\dot{\mathbf{K}}(t) = F(t, \mathbf{K}(t)\,\mathbf{V}^{0,\top})\,\mathbf{V}^0, \qquad \mathbf{K}(t_0) = \mathbf{U}^0\,\mathbf{S}^0$$

   and orthogonalizing $\mathbf{K}(t_1) = \mathbf{U}^1\,\widehat{\mathbf{S}}^1$

2. **S-step**: Update $\widehat{\mathbf{S}}^1 \to \widetilde{\mathbf{S}}^0$ by solving

$$\dot{\mathbf{S}}(t) = -\,\mathbf{U}^{1,\top}\,F(t, \mathbf{U}^1\,\mathbf{S}(t)\,\mathbf{V}^{0,\top})\,\mathbf{V}^0, \qquad \mathbf{S}(t_0) = \widehat{\mathbf{S}}^1$$

3. **L-step**: Update $\mathbf{V}^0 \to \mathbf{V}^1$, $\widetilde{\mathbf{S}}^0 \to \mathbf{S}^1$ by solving

$$\dot{\mathbf{L}}(t)^\top = \mathbf{U}^{1,\top}\,F(t, \mathbf{U}^1\,\mathbf{L}(t)^\top), \qquad \mathbf{L}(t_0)^\top = \widetilde{\mathbf{S}}^0\,\mathbf{V}^{0,\top}$$

   and orthogonalizing $\mathbf{L}(t_1) = \mathbf{V}^1\,\mathbf{S}^{1,\top}$.

The orthogonalization in the **K**- and in the **L**- step can be performed by a QR decomposition or by a SVD, respectively. In the remainder of this thesis, we will use the QR decomposition for this computation - not only within this matrix projector-splitting integrator, but also within the nested Tucker integrator, see Section 4.2.

Finally, merging the updated factor matrices yields $\mathbf{Y}^1 = \mathbf{U}^1\,\mathbf{S}^1\,\mathbf{V}^{1,\top} \approx \mathbf{Y}(t_1)$. Note that this product in practice is never computed in order to avoid an expensive computational effort. Compared to solving the equivalent system (2.7)–(2.9), this is an improvement with respect to the computational cost and memory for the integration procedure, since $\mathbf{K} \in \mathbb{R}^{n_1 \times r}$, $\mathbf{S} \in \mathbb{R}^{r \times r}$ and $\mathbf{L} \in \mathbb{R}^{n_2 \times r}$ are matrices of smaller size.

To continue with integrating in time, we take $\mathbf{U}^1, \mathbf{S}^1$ and $\mathbf{V}^1$ as initial values to start the time integration from $t_1 \to t_2$, and analogously for the subsequent time steps. We depict one time step of the matrix projector-splitting integrator in the following figure:

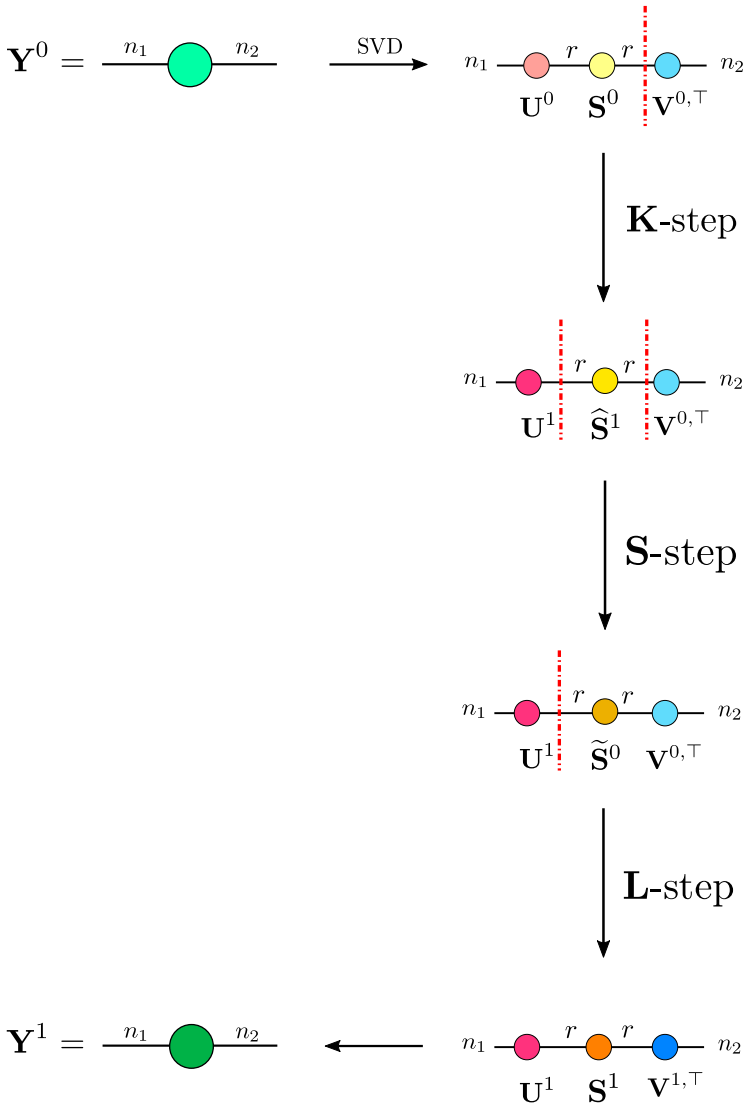

Figure 2.1: Schematic illustration of the matrix projector-splitting integration method.

We comment on the colors in the figure: the light colors depict the not yet updated matrices, i.e. $\mathbf{Y}^0$ is illustrated by the light green dot, the initial matrices $\mathbf{U}^0, \mathbf{S}^0, \mathbf{V}^0$ are depicted in light pink, light yellow and light blue, respectively. The updated basis matrices are then illustrated by the same, but darker, color. Since the matrix $\mathbf{S}$ is changed in each step of the projector-splitting integrator, the color changes in each step ranging from light yellow to deep orange. Finally, after having multiplied the three updated basis matrices, we obtain the approximation matrix $\mathbf{Y}^1$, which we depict in darker green.

Note that a first improvement of this integrator, compared to the method described in Section 1.3, can be seen here: we observe that the right-hand sides of the differential equations above are independent of the inverse of the matrix $\mathbf{S}$, which might contain small singular values.

Recall that the basis matrices $\mathbf{V}$ and $\mathbf{U}$ are not changed during the first two and the last two integration steps, respectively. They are not involved into the integration process

and so we can imagine to simply drop them in the correspondent step. The red dashed lines depict the cutting-off of the not yet or/and already updated matrices $\mathbf{V}$ or/and $\mathbf{U}$. The observation about the constant basis matrices in the appropriate steps within the integration scheme will play an important role in the error analysis of the projector-splitting integrator in Section 2.3. Beforehand, we will discuss this essential property in Section 2.2.2

2.1.2 Practical integration scheme

The implementation of the projector-splitting integrator follows directly from the derivation of the integration scheme. It requires solving the differential equations for $\mathbf{K}, \mathbf{S}$ and $\mathbf{L}^\top$ and additionally two QR decompositions to orthonormalize the columns of $\mathbf{K}$ and $\mathbf{L}$, respectively, in order to obtain orthonormal updated basis matrices $\mathbf{U}$ and $\mathbf{V}$. The description of the practical integration procedure for the time interval $[t_0, T]$ with time steps $(j-1)h = t_{j-1} \to t_j = jh$ for $j = 1, \dots, n$ and $t_n = T$ is given in Algorithm 1. It computes factors of a low-rank approximation $\mathbf{Y}^n = \mathbf{U}^n \mathbf{S}^n \mathbf{V}^{n,\top}$ after n time steps, which is taken as an approximation to $\mathbf{Y}(t_n)$.

Algorithm 1: Projector-splitting integrator

Data: Low-rank matrix $\mathbf{Y}^0 = \mathbf{U}^0 \mathbf{S}^0 \mathbf{V}^{0,\top}$, $F(t, \mathbf{Y})$, t_0, t_n, h
Result: Approximation matrix $\mathbf{Y}^n = \mathbf{U}^n \mathbf{S}^n \mathbf{V}^{n,\top}$

1 **begin**
2 **for** $j = 1$ **to** n **do**
3 set $\mathbf{K}^{j-1} = \mathbf{U}^{j-1} \mathbf{S}^{j-1}$
4 solve $\dot{\mathbf{K}}(t) = F\big(t, \mathbf{K}(t)\, \mathbf{V}^{j-1,\top}\big)\, \mathbf{V}^{j-1}$,
  with initial value $\mathbf{K}(t_{j-1}) = \mathbf{K}^{j-1}$ and return $\mathbf{K}^j = \mathbf{K}(t_j)$
5 compute QR factorization $\mathbf{K}^j = \mathbf{U}^j \widehat{\mathbf{S}}^j$
6 solve $\dot{\mathbf{S}}(t) = -\,\mathbf{U}^{j,\top} F\big(t, \mathbf{U}^j\, \mathbf{S}(t)\, \mathbf{V}^{j-1,\top}\big)\, \mathbf{V}^{j-1}$,
  with initial value $\mathbf{S}(t_{j-1}) = \widehat{\mathbf{S}}^j$ and return $\widetilde{\mathbf{S}}^{j-1} = \mathbf{S}(t_j)$
7 set $\mathbf{L}^{j-1,\top} = \widetilde{\mathbf{S}}^{j-1}\, \mathbf{V}^{j-1,\top}$
8 solve $\dot{\mathbf{L}}^{j,\top}(t) = \mathbf{U}^{j,\top} F\big(t, \mathbf{U}^j\, \mathbf{L}(t)^\top\big)$,
  with initial value $\mathbf{L}(t_{j-1})^\top = \mathbf{L}^{j-1,\top}$ and return $\mathbf{L}^{j,\top} = \mathbf{L}(t_j)^\top$
9 compute QR factorization $\mathbf{L}^j = \mathbf{V}^j\, \mathbf{S}^{j,\top}$
10 set $\mathbf{Y}^n = \mathbf{U}^n \mathbf{S}^n \mathbf{V}^{n,\top}$

The differential equations in lines 4, 6 and 8 need to be solved numerically and we therefore apply approximation methods such as a Runge–Kutta method. In case when F is independent of the time t and linear in $\mathbf{Y}$, we can apply a Krylov subspace method for computing the action of a matrix exponential, see [HL97, Saa92].

Note that in most applications, the low-rank initial value $\mathbf{Y}^0$ is not known beforehand. So in order to initialize the algorithm, we first perform a truncated SVD as described in Section 1.1, since we know by the Theorem of Eckart and Young that it results in a best approximation to the given $\mathbf{A}^0$.

Moreover, in case when F is solution independent, we are in the situation of projecting $\dot{\mathbf{A}}(t)$ onto the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$. This implies that the right-hand sides of the differential equations for $\mathbf{K}, \mathbf{S}$ and $\mathbf{L}$ in the scheme described in Section 2.1.1 also do not depend on the solution. Therefore, we can simply determine closed-form solutions to the three substeps, which for one time step $t_0 \to t_1$ are given by

$$\begin{aligned}
\mathbf{K}(t_1) &= \mathbf{K}(t_0) + \Big(\int_{t_0}^{t_1} \dot{\mathbf{A}}(s)ds\Big)\,\mathbf{V}(t_0) = \mathbf{K}(t_0) + \big(\mathbf{A}(t_1) - \mathbf{A}(t_0)\big)\,\mathbf{V}(t_0),\\
\mathbf{S}(t_1) &= \mathbf{S}(t_0) - \mathbf{U}(t_1)^\top\Big(\int_{t_0}^{t_1} \dot{\mathbf{A}}(s)ds\Big)\,\mathbf{V}(t_0) = \mathbf{S}(t_0) - \mathbf{U}(t_1)^\top\big(\mathbf{A}(t_1) - \mathbf{A}(t_0)\big)\,\mathbf{V}(t_0),\\
\mathbf{L}(t_1)^\top &= \mathbf{L}(t_0)^\top + \mathbf{U}(t_1)^\top\Big(\int_{t_0}^{t_1} \dot{\mathbf{A}}(s)ds\Big) = \mathbf{L}(t_0)^\top + \mathbf{U}(t_1)^\top\big(\mathbf{A}(t_1) - \mathbf{A}(t_0)\big).
\end{aligned} \tag{2.13}$$

This simplification of course also reduces the computational complexity, since there is no differential equation to solve. The corresponding practical integration is given in Algorithm 2, where we denote $\Delta\,\mathbf{A}^j = \mathbf{A}(t_j) - \mathbf{A}(t_{j-1})$, for $j = 1, \dots, n$. The algorithm then just uses the increment $\Delta\,\mathbf{A}^j$ instead of the time derivative $\dot{\mathbf{A}}(t)$.

Algorithm 2: Projector-splitting integrator, explicit case

Data: Low-rank matrix $\mathbf{Y}^0 = \mathbf{U}^0\,\mathbf{S}^0\,\mathbf{V}^{0,\top}$, $F(t, \mathbf{Y})$, t_0, t_n
Result: Approximation matrix $\mathbf{Y}^n = \mathbf{U}^n\,\mathbf{S}^n\,\mathbf{V}^{n,\top}$

begin
  for $j = 1$ **to** n **do**
    set $\mathbf{K}^{j-1} = \mathbf{U}^{j-1}\,\mathbf{S}^{j-1}$
    set $\mathbf{K}^j = \mathbf{K}^{j-1} + \Delta\,\mathbf{A}^j\,\mathbf{V}^{j-1}$
    compute QR factorization $\mathbf{K}^j = \mathbf{U}^j\,\widehat{\mathbf{S}}^j$
    set $\widetilde{\mathbf{S}}^{j-1} = \widehat{\mathbf{S}}^j - \mathbf{U}^{j,\top}\,\Delta\,\mathbf{A}^j\,\mathbf{V}^{j-1}$
    set $\mathbf{L}^{j-1,\top} = \widetilde{\mathbf{S}}^{j-1}\,\mathbf{V}^{j-1,\top}$
    set $\mathbf{L}^{j,\top} = \mathbf{L}^{j-1,\top} + \mathbf{U}^{j,\top}\,\Delta\,\mathbf{A}^j$
    compute QR factorization $\mathbf{L}^j = \mathbf{V}^j\,\mathbf{S}^{j,\top}$
  set $\mathbf{Y}^n = \mathbf{U}^n\,\mathbf{S}^n\,\mathbf{V}^{n,\top}$

2.2 Two substantial properties of the integrator

We devote this section to two properties, which will play an essential role in the error analysis of the matrix projector-splitting integrator. We start with discussing the remarkable exactness property, which was proven in [LO14, Theorem 4.1]. We also give a new proof that gives more insight why this surprising result holds. Afterwards, we will point out the property of constant projectors during the integration procedure, which was already briefly mentioned in the derivation of the scheme.

2.2.1 Exactness

We are accustomed to the fact that applying a numerical integrator to a given differential equation results in an approximate solution for which we can perform an error analysis that gives error bounds for the considered numerical integration method. Since this is the usual expectation with regard to a numerical solver, the exactness property of the projector-splitting integrator is astonishing.

In case when the matrix $\mathbf{A}(t)$ is not given implicitly as the solution to a given differential equation, as considered in the derivation of the integrator, but it is given explicitly, the differential equation for the low-rank approximation matrix is given by

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y}(t))\dot{\mathbf{A}}(t), \qquad \mathbf{A}(t_0) = \mathbf{A}^0\,. \tag{2.14}$$

Further, let the matrix $\mathbf{A}(t) \in \mathcal{M}$ be of low rank r for $t_0 \leq t \leq t_1$ and the initial value $\mathbf{A}^0$ be equal to the starting value $\mathbf{Y}^0$ of the algorithm. Then, by the additional technical assumption that a product of the coranges of $\mathbf{A}(t_1)$ and $\mathbf{Y}(t_0)$, respectively, is invertible, we conclude that the numerical solution of the matrix projector-splitting integrator applied to (2.14) with $\mathbf{A}^0 = \mathbf{Y}^0$ reproduces the exact matrix $\mathbf{A}(t)$ in the time grid points, i.e.,

$$\mathbf{Y}^1 = \mathbf{A}(t_1).$$

Note that the assumption about the non-singularity of the product of the $\mathbf{V}$-matrices is not present in the prerequisites in [LO14, Theorem 4.1]. However, going through their proof, it is obvious that this assumption is needed there.

The proof of [LO14, Theorem 4.1] simply goes through the substeps from bottom to top, i.e. from the **L**-step to the **K**-step and suddenly ends up with the exactness. In contrast, we will execute a new proof, which gives more insight into the appearing projections. We aim to give an improved explanation about where the exactness property comes from.

Theorem 2.1 (Exactness of the projector-splitting integrator)**.** *Suppose that $\mathbf{A}(t) \in \mathcal{M}$ for all $t_0 \leq t \leq T$, $\mathbf{A}(t_0) = \mathbf{Y}(t_0)$ and that the product $\mathbf{V}_{\mathbf{A}}(t_1)^\top \mathbf{V}(t_0)$ of the coranges of $\mathbf{A}(t_1)$ and $\mathbf{Y}(t_0)$, respectively, is invertible. Then, the Lie–Trotter projector-splitting integrator is exact, i.e. $\mathbf{Y}^1 = \mathbf{A}(t_1)$, for any $t_1 \in [t_0, T]$.*

In order to show this exactness property, we first show that in case when $\mathbf{A}(t)$ is of low rank, the solutions of the subproblems within the integrator are simply projections of $\mathbf{A}(t_1)$ and $\mathbf{A}(t_0)$, respectively. We formulate this observation in the following key lemma.

Lemma 2.2. *Let $\mathbf{A}(t) \in \mathcal{M}$ for all $t_0 \leq t \leq T$ and let $\mathbf{A}(t_0) = \mathbf{Y}(t_0)$. Then the solutions of the first two substeps of the projector-splitting integrator at $t_1 \in [t_0, T]$ are of the form*

$$\mathbf{Y}_1^+(t_1) = \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\,\mathbf{A}(t_1), \qquad \textit{and} \qquad \mathbf{Y}_1^-(t_1) = \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\,\mathbf{A}(t_0).$$

Proof. Since $\dot{\mathbf{A}}(t)$ is known explicitly and the right-hand side of the first subproblem within the **K**-step is solution independent, see (2.13), we can formulate a closed-form solution as

$$\mathbf{Y}_1^+(t_1) = \mathbf{Y}_1^+(t_0) + \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\Delta\,\mathbf{A}\,.$$

To start the integrator, let us denote $\mathbf{Y}_1^+(t_0) = \mathbf{Y}(t_0)$. By assumption, it holds that $\mathbf{A}(t_0) = \mathbf{Y}(t_0) = \mathbf{U}(t_0)\mathbf{S}(t_0)\,\mathbf{V}(t_0)^\top$ and hence we observe

$$\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\,\mathbf{A}(t_0) = \mathbf{A}(t_0)\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top = \mathbf{A}(t_0) = \mathbf{Y}_1^+(t_0).$$

This can be also seen by the fact that P_1^+ projects onto the corange of $\mathbf{Y}_1^+(t_0)$, but $\mathbf{Y}_1^+(t_0)$ and $\mathbf{A}(t_0)$ have the same corange. Hence, in fact, projecting $\mathbf{A}(t_0)$ by $\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))$ has no impact on $\mathbf{A}(t_0)$. Therefore, it follows for the solution of the first subproblem of the integrator that

$$\begin{aligned}\mathbf{Y}_1^+(t_1) &= \mathbf{Y}_1^+(t_0) + \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\,\mathbf{A}(t_1) - \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\,\mathbf{A}(t_0)\\ &= \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\,\mathbf{A}(t_1).\end{aligned} \tag{2.15}$$

The solution of the second subproblem can be expressed in closed form as

$$\mathbf{Y}_1^-(t_1) = \mathbf{Y}_1^-(t_0) - \mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))\Delta\,\mathbf{A}\,.$$

The splitting integrator is characterized by choosing the solution of the previous substep as initial value for the current approximation step. So by means of (2.15), we observe that the initial value for the second substep can be written in terms of $\mathbf{A}(t_1)$ as

$$\mathbf{Y}_1^-(t_0) = \mathbf{Y}_1^+(t_1) = \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\,\mathbf{A}(t_1). \tag{2.16}$$

Furthermore, by assumption $\mathbf{A}(t) \in \mathcal{M}$ for all considered times t, and so we can perform a SVD such as $\mathbf{A}(t_1) = \mathbf{U}_\mathbf{A}(t_1)\mathbf{S}_\mathbf{A}(t_1)\,\mathbf{V}_\mathbf{A}(t_1)^\top$. Then, by using (2.15), we obtain

$$\begin{aligned}\mathbf{U}(t_1)\mathbf{S}(t_1)\,\mathbf{V}(t_0)^\top &= \mathbf{Y}_1^+(t_1) = \mathbf{A}(t_1)\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top\\ &= \mathbf{U}_\mathbf{A}(t_1)\mathbf{S}_\mathbf{A}(t_1)\,\mathbf{V}_\mathbf{A}(t_1)^\top\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top.\end{aligned}$$

Now, by assumption, the product $\mathbf{V}_\mathbf{A}(t_1)^\top\,\mathbf{V}(t_0)$ is invertible and therefore we conclude that the range of $\mathbf{A}(t_1)$, which is given by $\mathbf{U}_\mathbf{A}(t_1)$, lies in the range of the updated basis matrix $\mathbf{U}(t_1)$ determined by the projector-splitting integrator. Therefore, projecting $\mathbf{A}(t_1)$ onto the space spanned by the range of $\mathbf{U}(t_1)$ in fact does not move $\mathbf{A}(t_1)$ on the low-rank manifold $\mathcal{M}$, i.e.,

$$\mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\,\mathbf{A}(t_1) = \mathbf{A}(t_1). \tag{2.17}$$

Denoting the composition of two projections by $\circ$, we therefore have

$$\begin{aligned}\mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))\,\mathbf{A}(t_1) = \mathrm{P}_1^-(\mathbf{Y}_1^+(t_1))\,\mathbf{A}(t_1) &= \big(\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0)) \circ \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\big)\,\mathbf{A}(t_1)\\ &= \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\,\mathbf{A}(t_1),\end{aligned}$$

which is why we conclude with (2.16) that $\mathbf{Y}_1^-(t_0) = \mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))\,\mathbf{A}(t_1)$. Projecting $\mathbf{A}(t_0)$ onto the space spanned by the range and corange of the initial value $\mathbf{Y}_1^-(t_0)$ of the second subproblem is the same as solely projecting it onto its range, i.e.,

$$\mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))\,\mathbf{A}(t_0) = \big(\mathrm{P}_2^+(\mathbf{Y}_1^+(t_1)) \circ \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\big)\,\mathbf{A}(t_0) = \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\,\mathbf{A}(t_0).$$

This can also be seen by the fact that the corange of $\mathbf{A}(t_0)$ and of $\mathbf{Y}_1^-(t_0)$ or $\mathbf{Y}_1^+(t_0)$, respectively, are the same, so projecting does not change $\mathbf{A}(t_0)$.

Therefore, we finally conclude that the solution of the second substep of the integrator is given by

$$\begin{aligned}\mathbf{Y}_1^-(t_1) &= \mathbf{Y}_1^-(t_0) - \mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))\,\mathbf{A}(t_1) + \mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))\,\mathbf{A}(t_0)\\ &= \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\,\mathbf{A}(t_0),\end{aligned}$$

which completes the proof. □

With the key lemma at hand, we now prove the exactness property of the matrix integrator.

Proof of Theorem 2.1. The approximation to $\mathbf{A}(t_1)$ after one time step of the projector-splitting integrator is determined by the solution of the third subproblem within the integration scheme, which is given by

$$\mathbf{Y}^1 = \mathbf{Y}_2^+(t_1) = \mathbf{Y}_2^+(t_0) + \mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))\Delta\,\mathbf{A}\,.$$

Now, since the range of $\mathbf{Y}_1^+(t_1)$ and of $\mathbf{Y}_2^+(t_0)$ is the same, the projection P_2^+ at those two different approximations maps onto the same space spanned by their range. Using the key Lemma 2.2, we can therefore rewrite the initial value of the third subproblem as

$$\mathbf{Y}_2^+(t_0) = \mathbf{Y}_1^-(t_1) = \mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))\,\mathbf{A}(t_0).$$

Since P_2^+ is a range-projection and due to the fact that the range of the update $\mathbf{U}(t_1)$ contains the range of the given $\mathbf{A}(t_1)$, see (2.17), we finally obtain

$$\mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))\,\mathbf{A}(t_1) = \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\,\mathbf{A}(t_1) = \mathbf{A}(t_1),$$

and so we conclude

$$\mathbf{Y}^1 = \mathbf{Y}_2^+(t_0) + \mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))\,\mathbf{A}(t_1) - \mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))\,\mathbf{A}(t_0) = \mathbf{A}(t_1).$$

□

Note that this result can be shown in an analogous way for the Strang splitting scheme, since latter consists of the same subprojections $\mathrm{P}_1^+, \mathrm{P}_1^-, \mathrm{P}_2^+$.

A generalization of this exactness theorem as well as of the key lemma for the higher-dimensional case will be given and proven in Chapter 4, where we will deal with the Tucker tensor format.

2.2.2 Constant projections

We have seen in the derivation of the matrix projector-splitting integrator that the method splits differential equation (2.2) into three subproblems. Within this projector-splitting integrator, we actually solve a system of two differential equations, see (2.10)-(2.12), in each step. There, we keep

- $\mathbf{V}^0$ constant when updating $\mathbf{U}^0 \to \mathbf{U}^1$ and $\mathbf{S}^0 \to \widehat{\mathbf{S}}^1$ in the **K**-step,
- $\mathbf{V}^0$ and $\mathbf{U}^1$ constant when updating $\widehat{\mathbf{S}}^1 \to \widetilde{\mathbf{S}}^0$ in the **S**-step,
- $\mathbf{U}^1$ constant when updating $\mathbf{V}^0 \to \mathbf{V}^1$ and $\widetilde{\mathbf{S}}^0 \to \mathbf{S}^1$ in the **L**-step.

Therefore, the solutions to the subproblems, starting with $\mathbf{Y}^0 = \mathbf{Y}_1^+(t_0) = \mathbf{U}^0\,\mathbf{S}^0\,\mathbf{V}^{0,\top}$ are given by:

- **K**-step: $\mathbf{Y}_1^+(t_0) = \mathbf{U}^0\,\mathbf{S}^0\,\mathbf{V}^{0,\top} \to \mathbf{U}^1\,\widehat{\mathbf{S}}^1\,\mathbf{V}^{0,\top} = \mathbf{Y}_1^+(t_1)$,
- **S**-step: $\mathbf{Y}_1^-(t_0) = \mathbf{U}^1\,\widehat{\mathbf{S}}^1\,\mathbf{V}^{0,\top} \to \mathbf{U}^1\,\widetilde{\mathbf{S}}^0\,\mathbf{V}^{0,\top} = \mathbf{Y}_1^-(t_1)$,
- **L**-step: $\mathbf{Y}_2^+(t_0) = \mathbf{U}^1\,\widetilde{\mathbf{S}}^0\,\mathbf{V}^{0,\top} \to \mathbf{U}^1\,\mathbf{S}^1\,\mathbf{V}^{1,\top} = \mathbf{Y}_2^+(t_1)$,

which is also illustrated in Figure 2.1. We observe that integration of the first subproblem does not change the corange of $\mathbf{Y}_1^+$, solving the second subproblem keeps both, the range and the corange of $\mathbf{Y}_1^-$ and integrating the last subproblem does not affect the range of $\mathbf{Y}_2^+$.

With regard to the projections that appear in each subproblem, we conclude that they are constant during the integration process, since they are composed of the basis matrices $\mathbf{U}$ and $\mathbf{V}$:

$$\begin{aligned}
&\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))F(t,\mathbf{Y}_1^+) = F(t,\mathbf{Y}_1^+)\,\mathbf{V}^0\,\mathbf{V}^{0,\top},\\
&\mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))F(t,\mathbf{Y}_1^-) = \mathbf{U}^1\,\mathbf{U}^{1,\top}\,F(t,\mathbf{Y}_1^-)\,\mathbf{V}^0\,\mathbf{V}^{0,\top},\\
&\mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))F(t,\mathbf{Y}_2^+) = \mathbf{U}^1\,\mathbf{U}^{1,\top}\,F(t,\mathbf{Y}_2^+).
\end{aligned}$$

In other words, by projecting F onto the corange and/or range of $\mathbf{Y}$, its corange and/or range are/is fixed. Then, performing a time integration "in the direction" of $\mathbf{U}$ and/or $\mathbf{V}$, we only update the factor matrices of $\mathbf{Y}$ that are not fixed.

For ease of understanding, we depict this property of the integrator in Figure 2.2. As an example, the low-rank manifold is to be viewed as a hyperboloid, which is a ruled surface and hence is constructed by straight lines in two different directions. In our setting, we think of those lines as being $\mathbf{U}$- and $\mathbf{V}$-directions. Starting from $\mathbf{Y}_1^+(t_0) = \mathbf{Y}_1^{+,0}$, which consists of $\mathbf{U}^0$ and $\mathbf{V}^0$, amongst others, we fix the direction of the corange and walk, i.e. integrate in time, into the direction of $\mathbf{U}$ on the blue $\mathbf{V}^0$-line in Figure 2.2, such that we end up on another crossing point $\mathbf{Y}_1^+(t_1)$ on the hyperboloid. Next, since range and corange of $\mathbf{Y}_1^+(t_1)$ are not changed during the integration, we seemingly do not move in any direction due to the low dimension of the exemplary ruled surface in Figure 2.2. Finally, when updating $\mathbf{V}$, we walk on the straight $\mathbf{U}^1$-line, depicted in pink in Figure 2.2, until we hit a $\mathbf{V}$-line, which then gives the updated basis matrix $\mathbf{V}^1$ and eventually end up at $\mathbf{Y}^1 = \mathbf{Y}_2^+(t_1)$.

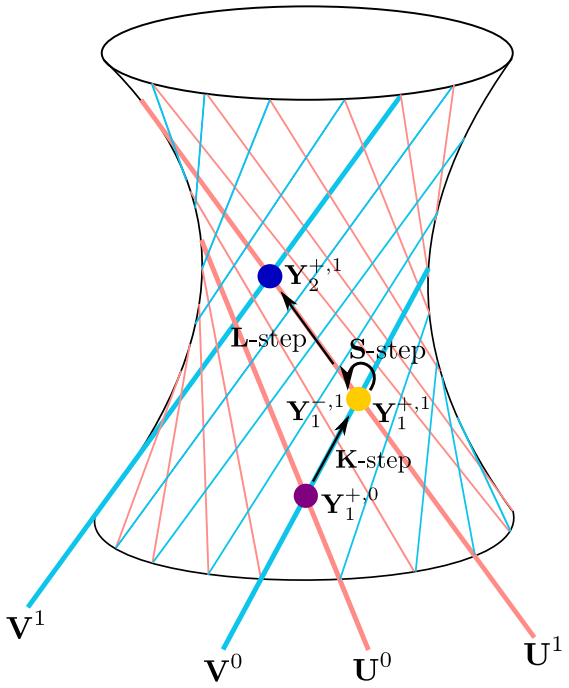

Figure 2.2: The integrator walks on straight lines.

This property of keeping the corange and/or range of the approximation matrix makes plausible why the matrix projector-splitting integrator does not suffer from small singular values. Suppose, the hyperboloid has a high curvature in the region of the saddle. By [KL07], this reflects small singular values of the current approximation matrix $\mathbf{Y}_i^{\pm}$. Now, since the integration of each substep within the projector-splitting integrator is performed on those straight lines, this integrator does not "see" the (high) curvature of the manifold, just because of keeping the corange and/or range within the integration procedure.

This is a crucial property for the error analysis of the matrix integrator, which will be treated next.

2.3 Robustness of the projector-splitting integrator with respect to small singular values

After having presented the integrator and discussed its beneficial properties, we will now turn to the main result of this chapter. We perform an error analysis, where the resulting error bound is independent of singular values, whether they are small or large. The two main ingredients for proving robustness of the integrator are the exactness property given in Section 2.2.1 and the projections that stay constant during each integration step, see Section 2.2.2. In principle, we will follow [KLW16, Theorem 2.1, Lemma 2.2], but beforehand, we give a clear structure of the proof and point out its strategy. The corresponding proofs of Theorem 2.4 and Lemma 2.5 then are broken down in detail and in a partly modified way.

Assumption 2.3. *We assume that*

(1) F is Lipschitz continuous:

$$\|F(t,\mathbf{Y}) - F(t,\widetilde{\mathbf{Y}})\| \leq L\|\mathbf{Y} - \widetilde{\mathbf{Y}}\| \qquad \forall\ \mathbf{Y},\widetilde{\mathbf{Y}} \in \mathbb{R}^{n_1\times n_2},$$

(2) F is bounded:

$$\|F(t,\mathbf{Y})\| \leq B \qquad \forall\ \mathbf{Y} \in \mathbb{R}^{n_1\times n_2},$$

(3) F is in the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$ up to a small perturbation term:

$$F(t,\mathbf{Y}) = M(t,\mathbf{Y}) + R(t,\mathbf{Y}),$$

where M maps to the tangent bundle of the low-rank manifold $\mathcal{M}$ and the remainder R is small on $\mathcal{M}$,

$$M(t,\mathbf{Y}) \in \mathcal{T}_{\mathbf{Y}}\mathcal{M} \quad \text{and} \quad \|R(t,\mathbf{Y})\| \leq \varepsilon \quad \forall\ \mathbf{Y} \in \mathcal{M}, \quad \forall\ t_0 \leq t \leq T,$$

(4) the initial value $\mathbf{A}^0 \in \mathbb{R}^{n_1\times n_2}$ and the starting value $\mathbf{Y}^0 \in \mathcal{M}$ of the numerical method are δ-close:

$$\|\mathbf{A}^0 - \mathbf{Y}^0\| \leq \delta.$$

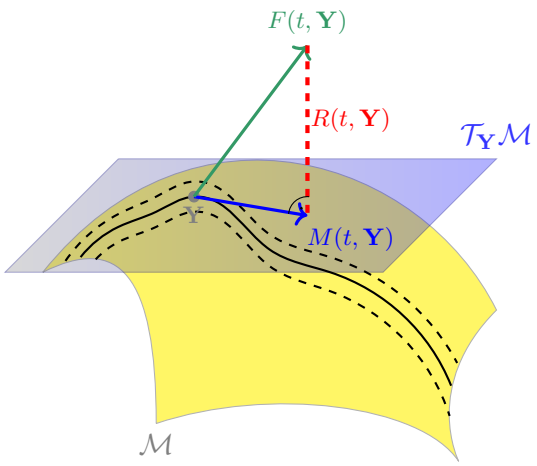

Figure 2.3: The situation, in which we give the convergence analysis of the projector-splitting integrator. The blue arrow depicts the projected $F(t,\mathbf{Y})$, which we imagine to be the tangential part $M(t,\mathbf{Y})$. Adding the time-dependent matrix $R(t,\mathbf{Y})$ to $M(t,\mathbf{Y})$ results in a sum, which is out of the tangent space. Within the figure, the red dashed line seems to be far away from the low-rank manifold, but in fact, by assumption, the perturbation term is controlled by ε. The black dashed line depicts the region around the trajectory (black solid line), in which we need the decomposition *(3)* of $F(t,\mathbf{Y})$.

We will take a closer look on the situation for which we analyze the error of the dynamical low-rank approximation by discussing the single items in the assumption.

(1) & (2) As usual in the numerical analysis of ordinary differential equations, those two assumptions could be weakened to a local Lipschitz condition and a local bound in a neighborhood of the exact solution of (2.1), where we denote its solution operator by $\mathbf{A}(t) = \Phi_F(t, t_0, \mathbf{A}^0)$, respectively. For convenience we will work with a global Lipschitz condition and bound.

(3) This assumption is needed along the solution trajectory $\{\mathbf{Y}(t) : t_0 \leq t \leq T\} \subset \mathcal{M}$ in order to obtain an approximation error $\mathbf{Y}(t) - \mathbf{A}(t) = \mathcal{O}(\varepsilon)$ for the time-continuous dynamical low-rank approximation $\mathbf{Y}(t) \in \mathcal{M}$. It is reasonable to make this assumption in a neighborhood on $\mathcal{M}$ of the trajectory. For convenience only, this assumption is made here for *all* $\mathbf{Y} \in \mathcal{M}$, but we would obtain the same result if we impose the assumption only in a small neighborhood on $\mathcal{M}$ of the trajectory.

The obvious choice for the decomposition and the way we imagine the tangential part of F is $M(t, \mathbf{Y}) = \mathrm{P}(\mathbf{Y})F(t, \mathbf{Y}) \in \mathcal{T}_{\mathbf{Y}}\mathcal{M}$, where again $\mathrm{P}(\mathbf{Y})$ denotes the orthogonal projection onto the tangent space, see Figure 2.3. We will not use any Lipschitz bound for M, since this would involve a local Lipschitz constant of $\mathrm{P}(\mathbf{Y})$, which is inversely proportional to the smallest non-zero singular value of $\mathbf{Y}$ and can thus become arbitrarily large, see [KL07, Lemma 4.2]. By assumption *(2)* and the bound of the perturbation term $R(t, \mathbf{Y})$, we have $\|M(t, \mathbf{Y})\| = \|F(t, \mathbf{Y}) - R(t, \mathbf{Y})\| \leq B + \varepsilon$. For convenience, we want $M(t, \mathbf{Y})$ to be bounded by the same bound as for $F(t, \mathbf{Y})$. Therefore, we choose the bound B of $F(t, \mathbf{Y})$ large enough, such that

$$\|M(t, \mathbf{Y})\| \leq B \qquad \forall\ \mathbf{Y} \in \mathcal{M} \quad \text{and} \quad \forall\ t_0 \leq t \leq T.$$

(4) We will see in the error analysis that the error bound depends on this distance, which is why we require it in Assumption 2.3 for the main theorem of this chapter. It means that we cannot expect a good approximation result, if $\mathbf{A}(t_0)$ is far away from the low-rank manifold in terms of the Frobenius norm, i.e. if δ is large. In practice, this assumption can be realized by performing a SVD of the initial value $\mathbf{A}^0$, such as described in Section 1.1, which yields a best rank r approximation $\mathbf{Y}^0$.

We are now in the position to state the convergence result of the dynamical low-rank approximation when applying the projector-splitting integrator.

Theorem 2.4. *Under Assumption 2.3, the error of the Lie–Trotter and of the Strang splitting method at $t_n = t_0 + nh$, with step size $h > 0$, is bounded by*

$$\|\mathbf{A}(t_n) - \mathbf{Y}^n\| \leq c_0\delta + c_1\varepsilon + c_2 h \qquad \textit{for } t_n \leq T,$$

where the constants c_i only depend on L, B and T.

We give the proof for the Lie–Trotter projector-splitting integrator. In case when the Strang splitting is applied, this proof also holds, since the Strang splitting consists of the

Lie–Trotter splitting for twice half the time step. This surprising result is supported by numerical evidence given within a numerical example in Section 2.5.3. It shows an error behavior of order one of the Strang splitting integration scheme.

We show this error bound by portioning the proof into several parts. Given the differential equation

$$\dot{\mathbf{A}}(t) = F(t, \mathbf{A}(t)), \qquad \mathbf{A}(t_0) = \mathbf{A}^0, \tag{2.18}$$

where the inital value $\mathbf{A}^0 \in \mathbb{R}^{n_1 \times n_2}$ can be of full rank, we compare its solution, which is denoted by $\Phi_F(t, t_0, \mathbf{A}^0)$, with the solution of the problem when starting on the low-rank manifold, i.e.,

$$\dot{\mathbf{A}}(t) = F(t, \mathbf{A}(t)), \qquad \mathbf{A}(t_0) = \mathbf{Y}^0, \tag{2.19}$$

where $\mathbf{Y}^0 \in \mathcal{M}$ and the solution operator is given by $\Phi_F(t, t_0, \mathbf{Y}^0)$. Having brought the situation onto the low-rank manifold, we then apply the Lie–Trotter projector-splitting integrator and obtain after n time steps at time $t_n = t_0 + nh$ the approximate solution $\mathbf{Y}^n$. The analysis of the error $\Phi_F(t_n, t_0, \mathbf{Y}^0) - \mathbf{Y}^n$ of the projector-splitting integrator completes the error estimates of Theorem 2.4, such that we have

$$\|\mathbf{A}(t_n) - \mathbf{Y}^n\| \leq \|\Phi_F(t_n, t_0, \mathbf{A}^0) - \Phi_F(t_n, t_0, \mathbf{Y}^0)\| + \|\Phi_F(t_n, t_0, \mathbf{Y}^0) - \mathbf{Y}^n\|.$$

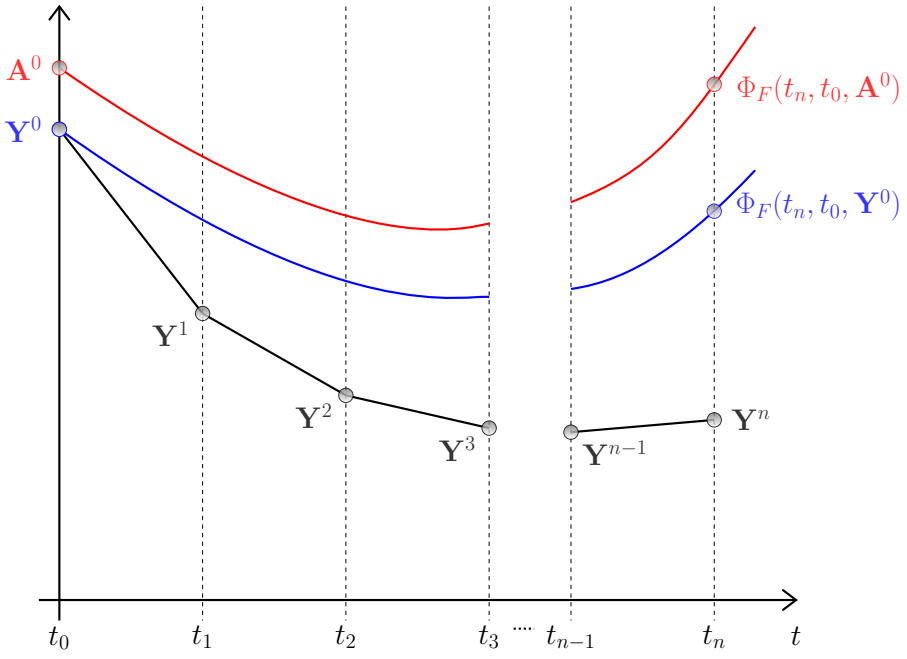

Figure 2.4: Schematic illustration of the convergence analysis. The uppermost curve (in red) depicts the exact solution $\mathbf{A}(t) = \Phi_F(t, t_0, \mathbf{A}^0)$ of (2.18), whereas the lowermost (in black) shows the solution obtained by the Lie–Trotter projector-splitting integrator. The line in between (in blue) represents the exact solution to the given problem (2.19) when starting with a low-rank initial value.

While the analysis of the propagation of the initial distance $\mathbf{A}^0 - \mathbf{Y}^0$ can be done by applying the Grönwall inequality, bounding the second term is not a straightforward task. We bound the error of the projector-splitting integrator by first analyzing its local error and then conclude the global error by propagating those local errors until the final time, see [HNW93, Lady Windermere's fan]. To this end, we consider the case, when $F(t, \mathbf{Y})$ already defines a vector field on the manifold, i.e., when the perturbation term $R(t, \mathbf{Y}) = 0$, and formulate the corresponding problem as

$$\dot{\mathbf{X}}(t) = M(t, \mathbf{X}(t)), \qquad \mathbf{X}(t_0) = \mathbf{X}^0, \tag{2.20}$$

where $\mathbf{X}^0 \in \mathcal{M}$. The solution to this problem is denoted by $\Phi_M(t, t_0, \mathbf{X}^0)$. With this auxiliary problem at hand, we bound the local error of the projector-splitting integrator by comparing the exact solution of the unperturbed problem (2.20) with the exact solution of (2.19) and with the numerical solution $\mathbf{Y}^k$ obtained by the projector-splitting integrator, for all $k = 0, \dots, n-1$, such that

$$\|\Phi_F(t_{k+1}, t_k, \mathbf{Y}^k) - \mathbf{Y}^{k+1}\| \leq \|\Phi_F(t_{k+1}, t_k, \mathbf{Y}^k) - \mathbf{X}(t_{k+1})\| + \|\mathbf{X}(t_{k+1}) - \mathbf{Y}^{k+1}\|.$$

The following figure renders a closer look of the strategy of showing this error estimate.

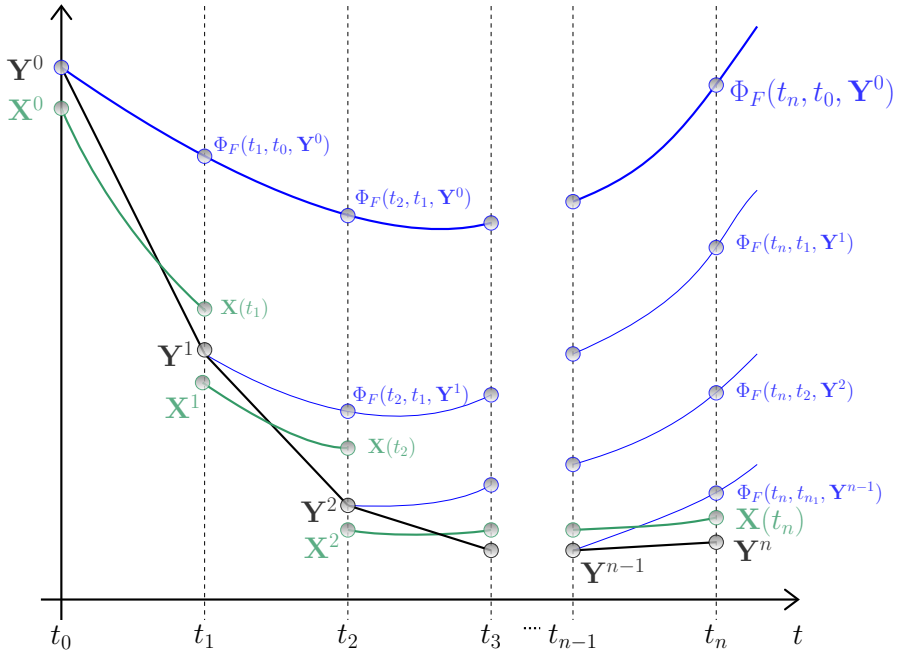

Figure 2.5: Schematic illustration of the convergence analysis of the projector-splitting integrator. The uppermost curve (in blue) depicts the exact solution $\Phi_F(t, t_0, \mathbf{Y}^0)$ of (2.19), when starting with a low-rank initial value. The lowermost line (in black) shows the numerical solution of (2.19) when applying the Lie–Trotter projector-splitting integrator. The curve in between (green) represents the exact solution of (2.20), which is the case when the right-hand side is a vector field, i.e. it is in the tangent space $\mathcal{T}_{\mathbf{X}}\mathcal{M}$.

The difficulty of the proof of Theorem 2.4 lies in estimating the local errors $\mathbf{X}(t_{k+1}) - \mathbf{Y}^{k+1}$. We formulate this result in the following lemma for the first time step $t_0 \to t_1$.

Lemma 2.5. *Let $\boldsymbol{X}^0 \in \mathcal{M}$ be the initial value of (2.20) with $\|\boldsymbol{X}^0 - \mathbf{Y}^0\| \leq h(4BLh + 2\varepsilon)$. Then, under Assumption 2.3 and with step size $h > 0$, there is the local error bound*

$$\|\boldsymbol{X}(t_1) - \mathbf{Y}^1\| \leq h(9BLh + 4\varepsilon),$$

where $t_1 = t_0 + h$.

Proof. To give the proof a clear structure, we will portion it out in several steps. In step 1, we will rewrite the differential equation (2.2) for the approximation matrix $\mathbf{Y}(t)$ in terms of a matrix $M(t, \mathbf{X}(t))$, which is in the tangent space of the low-rank manifold at the current approximation $\mathbf{Y}(t)$, plus a perturbation term $\Delta(t, \mathbf{Y}(t))$. This perturbation term appears in each subproblem of the projector-splitting integrator and therefore its effect needs to be investigated, which will be done in step 2. We analyze the influence of the perturbation term onto the solution in each subproblem in step 3. Due to the nature of the Lie–Trotter splitting scheme, the solution of each step is taken as initial value for the subsequent one. Hence, the perturbation term appears in the solution of the splitting integrator in nested form. We will show in step 4 that we can move the perturbation terms out of the splitting scheme, despite the nested form of the solution $\mathbf{Y}^1$. What comes to our help here, is the exactness property and the constant projections, which we have shown and explained in Sections 2.2.1 and 2.2.2, of the projector-splitting integrator. Finally, in step 5, we will bound the perturbation terms appearing in the substeps of the integration method.

First step: Rewriting the differential equation for $\mathbf{Y}$ in terms of $\dot{\mathbf{X}}(t)$.
Before considering the solution $\mathbf{Y}^1$ of the projector-splitting integrator, we first write the differential equation (2.2), to which we apply this integration method, in a way which facilitates the error analysis.

Following the third condition in Assumption 2.3, $M(t, \mathbf{Y}(t)) \in \mathcal{T}_Y\mathcal{M}$ and so

$$\mathrm{P}(\mathbf{Y}(t))M(t, \mathbf{Y}(t)) \in \mathcal{T}_Y\mathcal{M}.$$

Therefore, by inserting $\dot{\mathbf{X}}(t) = M(t, \mathbf{X}(t))$, we obtain

$$\begin{aligned}\dot{\mathbf{Y}}(t) &= \mathrm{P}(\mathbf{Y}(t))F(t, \mathbf{Y}(t)) = M(t, \mathbf{Y}(t)) + \mathrm{P}(\mathbf{Y}(t))R(t, \mathbf{Y}(t)) \\ &= \dot{\mathbf{X}}(t) - M(t, \mathbf{X}(t)) + M(t, \mathbf{Y}(t)) + \mathrm{P}(\mathbf{Y}(t))R(t, \mathbf{Y}(t)) \\ &= \dot{\mathbf{X}}(t) - F(t, \mathbf{X}(t)) + F(t, \mathbf{Y}(t)) \\ &\qquad + R(t, \mathbf{X}(t)) - R(t, \mathbf{Y}(t)) + \mathrm{P}(\mathbf{Y}(t))R(t, \mathbf{Y}(t)).\end{aligned} \tag{2.21}$$

Denoting

$$\Delta(t, \mathbf{Y}(t)) = F(t, \mathbf{Y}(t)) - F(t, \mathbf{X}(t)) - \big(\mathbf{I} - \mathrm{P}(\mathbf{Y}(t))\big)R(t, \mathbf{Y}(t)) + R(t, \mathbf{X}(t)),$$

we consider the evolution equation for $\mathbf{Y}(t)$ as an evolution equation for $\mathbf{X}(t)$ with a perturbation term $\Delta(t, \mathbf{Y})$ and so we have the differential equation

$$\dot{\mathbf{Y}}(t) = \dot{\mathbf{X}}(t) + \Delta(t, \mathbf{Y}(t)), \qquad \mathbf{Y}(t_0) = \mathbf{Y}^0\,. \tag{2.22}$$

By assumptions *(1)* and *(3)*, i.e., by the Lipschitz condition on F and the bound of R, the perturbation term is bounded by

$$\|\Delta(t, \mathbf{Y}(t))\| \leq L\|\mathbf{Y}(t) - \mathbf{X}(t)\| + 2\varepsilon. \tag{2.23}$$

We will use the fact that without the perturbation term $\Delta(t, \mathbf{Y}(t))$, the arising initial value problem

$$\dot{\widetilde{\mathbf{Y}}}(t) = \mathrm{P}(\widetilde{\mathbf{Y}}(t))\dot{\mathbf{X}}(t), \qquad \widetilde{\mathbf{Y}}(t_0) = \mathbf{X}(t_0)$$

with solution $\widetilde{\mathbf{Y}}(t) = \mathbf{X}(t)$ is solved exactly by the splitting integrator according to [LO14, Theorem 4.1] or Theorem 2.1 in Section 2.2.1. Setting $G_i^{\pm}(t, \mathbf{Y}) = \pm\mathrm{P}_i^{\pm}(\mathbf{Y})\dot{\mathbf{X}}(t)$, the exact solution of the projector-splitting integrator is given as

$$\mathbf{X}(t_1) = \Phi_{G_2^+}(t_1, t_0, \Phi_{G_1^-}(t_1, t_0, \Phi_{G_1^+}(t_1, t_0, \mathbf{X}(t_0)))).$$

Second step: Investigating the effect of the perturbation term in each substep.
Since the evolution equation (2.22) is solved numerically by the projector-splitting integrator, we have to take the subprojections, which appear in the splitting scheme, into account for our analysis. The subprojection P_1^- is a projection onto the range and corange of the current approximation matrix. Hence, it is a composition of the projections P_1^+ and P_2^+ and so we can write $\mathrm{P}_1^- = \mathrm{P}_2^+ \circ \mathrm{P}_1^+$. Now, combining an arbitrary but fix subprojection $\mathrm{P}_i^{\pm}$ with each P_1^+, P_1^- and P_2^+, changes either P_1^+ or P_2^+ or both into $\mathrm{P}_1^- = \mathrm{P}_2^+ \circ \mathrm{P}_1^+$, which then cancels with the already existing projection P_1^-. So we are left with

$$\begin{aligned}\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\mathrm{P}(\mathbf{Y}) &= \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\big(\mathrm{P}_1^+(\mathbf{Y}_1^+) - (\mathrm{P}_2^+ \circ \mathrm{P}_1^+)(\mathbf{Y}_1^+) + \mathrm{P}_2^+(\mathbf{Y}_2^+)\big)\\ &= \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm}).\end{aligned}$$

Therefore, we have

$$\begin{aligned}\dot{\mathbf{Y}}(t) &= \mathrm{P}(\mathbf{Y}(t))F(t, \mathbf{Y}(t))\\ &= \big(\mathrm{P}_1^+(\mathbf{Y}_1^+(t)) - \mathrm{P}_1^-(\mathbf{Y}_1^-(t)) + \mathrm{P}_2^+(\mathbf{Y}_2^+(t))\big)F(t, \mathbf{Y}(t))\\ &= \big(\mathrm{P}_1^+(\mathbf{Y}_1^+(t)) - \mathrm{P}_1^-(\mathbf{Y}_1^-(t)) + \mathrm{P}_2^+(\mathbf{Y}_2^+(t))\big)\mathrm{P}(\mathbf{Y})F(t, \mathbf{Y}(t))\\ &= \big(\mathrm{P}_1^+(\mathbf{Y}_1^+(t)) - \mathrm{P}_1^-(\mathbf{Y}_1^-(t)) + \mathrm{P}_2^+(\mathbf{Y}_2^+(t))\big)\big(\dot{\mathbf{X}}(t) + \Delta(t, \mathbf{Y}(t)\big),\end{aligned}$$

and so the differential equations solved in the substeps of the splitting integrator can be written as

$$\dot{\mathbf{Y}}_i^{\pm}(t) = \pm\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm}(t))\dot{\mathbf{X}}(t) \pm \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm}(t))\Delta(t, \mathbf{Y}_i^{\pm}(t)),$$

where $t_0 \leq t \leq t_0 + h$. Setting $\Delta_i^{\pm}(t, \mathbf{Y}) = \pm\mathrm{P}_i^{\pm}(\mathbf{Y})\Delta(t, \mathbf{Y})$ we have in abbreviated form

$$\dot{\mathbf{Y}}_i^{\pm}(t) = F_i^{\pm}(t, \mathbf{Y}_i^{\pm}(t)) = G_i^{\pm}(t, \mathbf{Y}_i^{\pm}(t)) + \Delta_i^{\pm}(t, \mathbf{Y}_i^{\pm}(t)), \tag{2.24}$$

with solution $\mathbf{Y}_i^{\pm}(t) = \Phi_{F_i^{\pm}}(t, t_0, \mathbf{Y}_i^{\pm}(t_0))$. For analyzing the effect of the perturbation term $\Delta(t, \mathbf{Y}(t))$ in (2.22) on the solution obtained by the projector-splitting integrator, we

compare the solution $\Phi_{F_i^\pm}(t, t_0, \mathbf{Y}_i^\pm(t_0))$ of the full subproblem (2.24) with the solution of the perturbation-free differential equation

$$\dot{\widetilde{\mathbf{Y}}}_i^\pm(t) = \mathrm{P}(\widetilde{\mathbf{Y}}_i^\pm(t))\dot{\mathbf{X}}(t) = G_i^\pm(t, \widetilde{\mathbf{Y}}_i^\pm(t)), \qquad \widetilde{\mathbf{Y}}_i^\pm(t_0) = \mathbf{Y}_i^\pm(t_0), \tag{2.25}$$

which is denoted by $\widetilde{\mathbf{Y}}_i^\pm(t) = \Phi_{G_i^\pm}(t, t_0, \widetilde{\mathbf{Y}}_i^\pm(t_0))$. Using the nonlinear variation-of-constants formula by W. Gröbner [Grö67] and V.M. Alekseev [Ale61] proposed independently from each other and brought together in [HNW93, Theorem I.14.5], we see how the solutions of the differential equations (2.24) and (2.25) are connected. Denoting the derivative of $\Phi_{G_i^\pm}(t_1, s, \mathbf{Y})$ with respect to the initial value $\mathbf{Y}$ by $\partial\Phi_{G_i^\pm}(t_1, s, \mathbf{Y})$ and with $\mathbf{Y}_i^\pm(s) = \Phi_{F_i^\pm}(s, t_0, \mathbf{Y}_i^\pm(t_0))$, we have, at $t_1 = t_0 + h$,

$$\begin{aligned}\Phi_{F_i^\pm}(t_1, t_0, \mathbf{Y}_i^\pm(t_0)) = \Phi_{G_i^\pm}(t_1, t_0, \mathbf{Y}_i^\pm(t_0)) \\ + \int_{t_0}^{t_1} \partial\Phi_{G_i^\pm}(t_1, s, \mathbf{Y}_i^\pm(s)) \cdot \Delta_i^\pm(s, \mathbf{Y}_i^\pm(s))\,\mathrm{d}s.\end{aligned} \tag{2.26}$$

The effect of the perturbation term on the solution in each substep of the projector-splitting integrator is described by the integrand.

Third step: Surveying the integrand.
We bound the integrand by considering it as a directional derivative, which describes the behavior of the solution for the unperturbed problem (2.25) in the direction of the perturbation term. We ask, how influential the perturbation term $\Delta_i^\pm(t, \mathbf{Y}_i^\pm(t))$ with respect to the solution $\widetilde{\mathbf{Y}}_i^\pm(t)$ is. To answer this question, we fix the direction by fixing the perturbation term in the sense that we consider it at an arbitrary but fix time $t_0 \le s \le t_1$ and vary the derivative of the flow $\Phi_{G_i^\pm}$ with respect to the initial value:

$$E_i^\pm(\tau, s) = \partial\Phi_{G_i^\pm}(t_1, \tau, \mathbf{Y}_i^\pm(s)) \cdot \Delta_i^\pm(s, \mathbf{Y}_i^\pm(s)).$$

This yields

$$\Phi_{F_i^\pm}(t_1, t_0, \mathbf{Y}_i^\pm(t_0)) = \Phi_{G_i^\pm}(t_1, t_0, \mathbf{Y}_i^\pm(t_0)) + \int_{t_0}^{t_1} E_i^\pm(\tau, s)\Big|_{\tau=s}\,\mathrm{d}s.$$

To emphasize the fact that s is fixed, we simply drop the dependence on s in the notation, such that $\mathbf{Y}_i^\pm(s)$ becomes $\mathbf{Y}_i^\pm$ and for the perturbation term we have $\Delta_i^\pm(s, \mathbf{Y}_i^\pm(s)) = \mathrm{P}_i^\pm(\mathbf{Y}_i^\pm)\Delta(s, \mathbf{Y}_i^\pm(s)) = \mathrm{P}_i^\pm(\mathbf{Y}_i^\pm)\Delta$ as a short form. Now, expressing the directional derivative of $\Phi_{G_i^\pm}$ with respect to the initial value by explicitly taking the limit yields

$$E_i^\pm(\tau) = \lim_{\theta\to 0} \frac{1}{\theta}\Big(\Phi_{G_i^\pm}(t_1, \tau, \mathbf{Y}_i^\pm + \theta\mathrm{P}_i^\pm(\mathbf{Y}_i^\pm)\Delta) - \Phi_{G_i^\pm}(t_1, \tau, \mathbf{Y}_i^\pm)\Big).$$

In the following, we show that $E_i^\pm(\tau)$ is actually independent of $\tau \in [t_0, t_1]$. To this end, we make use of the observation that the range and the corange of $\mathbf{Y}_i^\pm$ are preserved under the subflows $\Phi_{G_i^\pm}$ as discussed in Section 2.2.2, i.e.,

$$\mathrm{P}_i^\pm(\Phi_{G_i^\pm}(t_1, \tau, \mathbf{Y}_i^\pm)) = \mathrm{P}_i^\pm(\mathbf{Y}_i^\pm). \tag{2.27}$$

Moreover, the projections $\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})$ are invariant under adding a multiple of $\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta$ to the argument $\mathbf{Y}_i^{\pm}$:

$$\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm} + \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta) = \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm}). \tag{2.28}$$

Now, computing the derivative of $E_i^{\pm}(\tau)$ backwards in time with respect to τ yields

$$\begin{aligned}
\dot{E}_i^{\pm}(\tau) &= -\lim_{\theta\to 0}\frac{1}{\theta}\Big(G_i^{\pm}(\tau, \Phi_{G_i^{\pm}}(t_1,\tau,\mathbf{Y}_i^{\pm} + \theta\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta)) - G_i^{\pm}(\tau, \Phi_{G_i^{\pm}}(t_1,\tau,\mathbf{Y}_i^{\pm}))\Big)\\
&= \mp\lim_{\theta\to 0}\frac{1}{\theta}\Big(\mathrm{P}_i^{\pm}\big(\Phi_{G_i^{\pm}}(t_1,\tau,\mathbf{Y}_i^{\pm} + \theta\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta)\big)\dot{\mathbf{X}}(\tau) - \mathrm{P}_i^{\pm}(\Phi_{G_i^{\pm}}(t_1,\tau,\mathbf{Y}_i^{\pm}))\dot{\mathbf{X}}(\tau)\Big)\\
&\overset{(2.27)}{=} \mp\lim_{\theta\to 0}\frac{1}{\theta}\Big(\mathrm{P}_i^{\pm}\big(\mathbf{Y}_i^{\pm} + \theta\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta\big)\dot{\mathbf{X}}(\tau) - \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\dot{\mathbf{X}}(\tau)\Big)\\
&\overset{(2.28)}{=} \mp\lim_{\theta\to 0}\frac{1}{\theta}\big(\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\dot{\mathbf{X}}(\tau) - \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\dot{\mathbf{X}}(\tau)\big)\\
&= 0.
\end{aligned}$$

Hence, $E_i^{\pm}(\tau)$ in fact does not depend on τ with $t_0 \le \tau \le t_1$ and so we have

$$\begin{aligned}
E_i^{\pm}(\tau) &= \partial\Phi_{G_i^{\pm}}(t_1,\tau,\mathbf{Y}_i^{\pm})\cdot\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta\\
&= \partial\Phi_{G_i^{\pm}}(t_1,t_1,\mathbf{Y}_i^{\pm})\cdot\mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta\\
&= \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm})\Delta,
\end{aligned}$$

since $\partial\Phi_{G_i^{\pm}}(t_1,t_1,\mathbf{Y}_i^{\pm})$ is the identity matrix.

Therefore, in (2.26) we are left with

$$\begin{aligned}
\Phi_{F_i^{\pm}}(t_1,t_0,\mathbf{Y}_i^{\pm}(t_0)) &= \Phi_{G_i^{\pm}}(t_1,t_0,\mathbf{Y}_i^{\pm}(t_0)) + \int_{t_0}^{t_1} E_i^{\pm}(\tau,s)\Big|_{\tau=s}\,\mathrm{d}s\\
&= \Phi_{G_i^{\pm}}(t_1,t_0,\mathbf{Y}_i^{\pm}(t_0)) + \int_{t_0}^{t_1} \mathrm{P}_i^{\pm}(\mathbf{Y}_i^{\pm}(s))\Delta(s,\mathbf{Y}_i^{\pm}(s))\,\mathrm{d}s\\
&= \Phi_{G_i^{\pm}}(t_1,t_0,\mathbf{Y}_i^{\pm}(t_0)) + \int_{t_0}^{t_1} E_i^{\pm}(s)\,\mathrm{d}s.
\end{aligned}$$

For a convenient notation, we denote $E_i^{\pm} := \int_{t_0}^{t_1} E_i^{\pm}(s)\,\mathrm{d}s$, such that we obtain the result $\mathbf{Y}^1$ of one step of the splitting method as

$$\begin{aligned}
\mathbf{Y}^1 &= \Phi_{F_2^+}(t_1,t_0,\Phi_{F_1^-}(t_1,t_0,\Phi_{F_1^+}(t_1,t_0,\mathbf{Y}^0)))\\
&= \Phi_{G_2^+}(t_1,t_0,\Phi_{G_1^-}(t_1,t_0,\Phi_{G_1^+}(t_1,t_0,\mathbf{Y}^0) + E_1^+) + E_1^-) + E_2^+.
\end{aligned} \tag{2.29}$$

Fourth step: Moving the perturbation terms out of the splitting scheme.
Though having simplified the integrand, it is still not straightforward how to analyze the influence of the perturbation onto the solution of the unperturbed subproblems (2.25). The difficulty lies in the fact that the terms $E_i^{\pm}$ appear in nested form in the splitting scheme (2.29). What comes to our help is again the preservation of the range and corange of the approximation matrix by the solution operators $\Phi_{F_i^{\pm}}$ and $\Phi_{G_i^{\pm}}$, see Section 2.2.2. Using this crucial feature, we are able to move the error terms E_1^+ and E_1^- out of the splitting

scheme, such that we get rid of the nestedness. In the following we first list the substeps within the splitting scheme. Since $\mathbf{Y}_1^-(t_0) = \mathbf{Y}_1^+(t_1)$ and $\mathbf{Y}_2^+(t_0) = \mathbf{Y}_1^-(t_1)$, we have

$$\begin{aligned}
\mathbf{Y}_1^+(t_1) &= \Phi_{F_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0)) = \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0)) + E_1^+, \\
\mathbf{Y}_1^-(t_1) &= \Phi_{F_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1)) = \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1)) + E_1^-, \\
\mathbf{Y}^1 = \mathbf{Y}_2^+(t_1) &= \Phi_{F_2^+}(t_1, t_0, \mathbf{Y}_1^-(t_1)) = \Phi_{G_2^+}(t_1, t_0, \mathbf{Y}_1^-(t_1)) + E_2^+.
\end{aligned} \tag{2.30}$$

Since the solution operator $\Phi_{F_1^-}(t, t_0, \mathbf{Y}_1^+(t_1))$ preserves both the range and corange of $\mathbf{Y}_1^+(t_1)$, and since $\mathrm{P}_2^+(\mathbf{Y})$ is the projection onto the range of $\mathbf{Y}$, for $\mathbf{Y} \in \mathcal{M}$, we have

$$\mathrm{P}_2^+(\mathbf{Y}_1^-(t_1)) = \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1)).$$

Looking at (2.29), we observe that the error term E_2^+ of the solution of the third subproblem is not nested in the splitting scheme. Next, we show that solving the third subproblem in (2.25) when starting with a perturbed initial value $\mathbf{Y}_1^-(t_1) = \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1)) + E_1^-$, where E_1^- is the perturbation term here, gives the same result as when the initial value is unperturbed but the perturbation is added to the solution $\Phi_{G_2^+}(t_1, t_0, \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1))$. In other words, we show that the perturbation term E_1^- can be moved out of the splitting scheme. To this end, we consider how the difference of the two solutions, when starting with or without the perturbation term E_1^-, changes in time. This is the point where the range and corange preservation of the subflows plays a key role: since the solution operator $\Phi_{G_2^+}$ preserves the range and $\Phi_{G_1^-}$ preserves both, the range and the corange of the correspondent initial value, we find

$$\begin{aligned}
&\frac{\mathrm{d}}{\mathrm{d}t}\Big(\Phi_{G_2^+}(t, t_0, \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1)) + E_1^-) - \Phi_{G_2^+}(t, t_0, \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1)))\Big) \\
&= G_2^+\big(t, \Phi_{G_2^+}(t, t_0, \mathbf{Y}_1^-(t_1))\big) - G_2^+\Big(t, \Phi_{G_2^+}\big(t, t_0, \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1))\big)\Big) \\
&= \mathrm{P}_2^+\big(\Phi_{G_2^+}(t, t_0, \mathbf{Y}_1^-(t_1))\big)\dot{\mathbf{X}}(t) - \mathrm{P}_2^+\Big(\Phi_{G_2^+}\big(t, t_0, \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1))\big)\Big)\dot{\mathbf{X}}(t) \\
&= \mathrm{P}_2^+(\mathbf{Y}_1^-(t_1))\dot{\mathbf{X}}(t) - \mathrm{P}_2^+\big(\Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1))\big)\dot{\mathbf{X}}(t) \\
&= \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\dot{\mathbf{X}}(t) - \mathrm{P}_2^+(\mathbf{Y}_1^+(t_1))\dot{\mathbf{X}}(t) \\
&= 0,
\end{aligned}$$

for all $t_0 \le t \le t_1$. It follows that

$$\begin{aligned}
&\Phi_{G_2^+}\Big(t, t_0, \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big) + E_1^-\Big) - \Phi_{G_2^+}\Big(t, t_0, \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big)\Big) \\
&= \Phi_{G_2^+}\Big(t_0, t_0, \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big) + E_1^-\Big) - \Phi_{G_2^+}\Big(t_0, t_0, \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big)\Big) \\
&= \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big) + E_1^- - \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big) \\
&= E_1^-.
\end{aligned}$$

This is equivalent to

$$\Phi_{G_2^+}\Big(t, t_0, \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big) + E_1^-\Big) = \Phi_{G_2^+}\Big(t, t_0, \Phi_{G_1^-}\big(t_1, t_0, \mathbf{Y}_1^+(t_1)\big)\Big) + E_1^-.$$

Going back to the splitting scheme, this remarkable observation results in

$$\begin{aligned}\mathbf{Y}^1 &= \Phi_{G_2^+}(t_1, t_0, \mathbf{Y}_1^-(t_1)) + E_2^+ \\ &= \Phi_{G_2^+}\big(t_1, t_0, \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1)) + E_1^-\big) + E_2^+ \\ &= \Phi_{G_2^+}\big(t_1, t_0, \Phi_{G_1^-}(t_1, t_0, \mathbf{Y}_1^+(t_1))\big) + E_1^- + E_2^+.\end{aligned}$$

To get rid of the nestedness of the final error term E_1^+ within the splitting scheme, we show that the solution for the first subproblem when starting with $\mathbf{Y}_1^+(t_0)$ in (2.25) and then adding the perturbation E_1^+ to this solution is the same as taking $\mathbf{Y}_1^+(t_0) + E_1^+$ as initial value for (2.25). Due to the corange preservation, we observe that

$$\begin{aligned}\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))E_1^+ &= \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0)) \int_{t_0}^{t_1} \mathrm{P}_1^+(\mathbf{Y}_1^+(s))\Delta(s, \mathbf{Y}_1^+(s))\,\mathrm{d}s \\ &= \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0)) \int_{t_0}^{t_1} \Delta(s, \mathbf{Y}_1^+(s))\,\mathrm{d}s \\ &= \int_{t_0}^{t_1} \Delta(s, \mathbf{Y}_1^+(s))\,\mathrm{d}s \\ &= E_1^+.\end{aligned}$$

Further, we remind that the solution operator $\Phi_{G_1^+}$ preserves the corange of the initial value and that the projection $\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))$ is invariant under a multiple of $\mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))E_1^+$, see (2.28). This yields

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\Big(\Phi_{G_1^+}(t, t_0, \mathbf{Y}_1^+(t_0) + E_1^+) - \Phi_{G_1^+}(t, t_0, \mathbf{Y}_1^+(t_0))\Big) \\ &= G_1^+\big(t, \Phi_{G_1^+}(t, t_0, \mathbf{Y}_1^+(t_0) + E_1^+)\big) - G_1^+\big(t, \Phi_{G_1^+}(t, t_0, \mathbf{Y}_1^+(t_0))\big) \\ &= \mathrm{P}_1^+\big(\Phi_{G_1^+}(t, t_0, \mathbf{Y}_1^+(t_0) + E_1^+)\big)\dot{\mathbf{X}}(t) - \mathrm{P}_1^+\big(\Phi_{G_1^+}(t, t_0, \mathbf{Y}_1^+(t_0))\big)\dot{\mathbf{X}}(t) \\ &= \mathrm{P}_1^+\big(\mathbf{Y}_1^+(t_0) + E_1^+\big)\dot{\mathbf{X}}(t) - \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\dot{\mathbf{X}}(t) \\ &= \mathrm{P}_1^+\big(\mathbf{Y}_1^+(t_0) + \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))E_1^+\big)\dot{\mathbf{X}}(t) - \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\dot{\mathbf{X}}(t) \\ &= \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\dot{\mathbf{X}}(t) - \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\dot{\mathbf{X}}(t) \\ &= 0.\end{aligned}$$

Hence, the difference of the flow $\Phi_{G_1^+}$ when starting with $\mathbf{Y}_1^+(t_0) + E_1^+$ or with $\mathbf{Y}_1^+(t_0)$, respectively, does not change in time. This means that

$$\begin{aligned}\Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0) + E_1^+) - \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0)) &= (\mathbf{Y}_1^+(t_0) + E_1^+) - \mathbf{Y}_1^+(t_0) \\ &= E_1^+,\end{aligned}$$

which is equivalent to

$$\Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0) + E_1^+) = \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0)) + E_1^+. \tag{2.31}$$

Therefore, looking at the splitting scheme, we have

$$\begin{aligned}\mathbf{Y}^1 &= \Phi_{G_2^+}\Big(t_1, t_0, \Phi_{G_1^-}\big(t_1, t_0, \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0)) + E_1^+\big)\Big) + E_1^- + E_2^+ \\ &= \Phi_{G_2^+}\Big(t_1, t_0, \Phi_{G_1^-}\big(t_1, t_0, \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0) + E_1^+)\big)\Big) + E_1^- + E_2^+.\end{aligned}$$

Now, since $\mathbf{Y}_1^+(t_0)$ and E_1^+ have the same range, their sum is of rank at most r. If its rank is less than r, we can perturb it by a small term, which is of no consequence to our error bound and can be controlled by Lady Windermere's fan. Hence, the sum is of rank r and therefore we choose $\mathbf{X}(t_0) = \mathbf{Y}_1^+(t_0) + E_1^+$, where we obtain by the exactness result of [LO14, Theorem 4.1] and accordingly Theorem 2.1 in Section 2.2.1 the exact solution for the unperturbed problem, given by

$$\Phi_{G_2^+}(t_1, t_0, \Phi_{G_1^-}(t_1, t_0, \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}_1^+(t_0) + E_1^+))) = \mathbf{X}(t_1).$$

Therefore, the solution for (2.19) is reduced to

$$\mathbf{Y}^1 = \mathbf{X}(t_1) + E_1^- + E_2^+.$$

Fifth step: Bounding the perturbation terms.
In order to bound

$$\begin{aligned} E_i^\pm &= \pm \int_{t_0}^{t_1} \mathrm{P}_i^\pm(\mathbf{Y}_i^\pm(s))\Delta(s, \mathbf{Y}_i^\pm(s))\, ds \\ &= \pm \mathrm{P}_i^\pm(\mathbf{Y}_i^\pm(t_0)) \int_{t_0}^{t_1} \Delta(s, \mathbf{Y}_i^\pm(s))\, ds, \end{aligned} \tag{2.32}$$

we bound the perturbation term $\Delta(s, \mathbf{Y}_i^\pm(s))$. Using the estimate (2.23) for $\mathbf{Y}_i^\pm$, we have to determine the Frobenius norm of $\mathbf{Y}_i^\pm(s) - \mathbf{X}(s)$, where s plays the role of t and $\mathbf{Y}_i^\pm(t)$ solves

$$\dot{\mathbf{Y}}_i^\pm(t) = \mathrm{P}_i^\pm(\mathbf{Y}_i^\pm(t))F(t, \mathbf{Y}_i^\pm(t)) \tag{2.33}$$

with initial value $\mathbf{Y}_i^\pm(t_0)$ and $\mathbf{X}(t)$ is the solution of

$$\dot{\mathbf{X}}(t) = M(t, \mathbf{X}(t)) \tag{2.34}$$

with initial value $\mathbf{X}(t_0) = \mathbf{X}^0$. Then, by the bound B for $F(t, \mathbf{Y}_i^\pm(t))$ and $M(t, \mathbf{Y}_i^\pm(t))$ for all $\mathbf{Y}_i^\pm(t) \in \mathcal{M}$, we obtain by integrating (2.33) and (2.34) on both sides, respectively, the bounds

$$\|\mathbf{Y}_i^\pm(t_1) - \mathbf{Y}_i^\pm(t_0)\| \leq Bh \qquad \text{and} \qquad \|\mathbf{X}(t_1) - \mathbf{X}(t_0)\| \leq Bh.$$

We observe that for the first substep 1^+, we have with (2.30) and (2.31) that

$$\mathbf{Y}_1^+(t_1) = \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}^0) + E_1^+ = \Phi_{G_1^+}(t_1, t_0, \mathbf{X}^0),$$

where $\Phi_{G_1^+}(t_1, t_0, \mathbf{X}^0)$ is the solution of the unperturbed problem (2.25) for $i = 1$ and $+$, i.e.,

$$\dot{\widetilde{\mathbf{Y}}}_1^+(t) = \mathrm{P}(\widetilde{\mathbf{Y}}_1^+(t))\dot{\mathbf{X}}(t) = G_1^+(t, \widetilde{\mathbf{Y}}_1^+(t)), \qquad \widetilde{\mathbf{Y}}_1^+(t_0) = \mathbf{X}^0\,.$$

Now, integrating this differential equation on both sides, we obtain, again by the boundedness of $M(t, \mathbf{X}(t))$, the estimate

$$\|\Phi_{G_1^+}(t_1, t_0, \mathbf{X}^0) - \mathbf{X}^0\| \leq Bh.$$

Hence,

$$\|\mathbf{Y}_1^+(t_1) - \mathbf{X}(t_1)\| \leq \|\Phi_{G_1^+}(t_1, t_0, \mathbf{X}^0) - \mathbf{X}^0\| + \|\mathbf{X}(t_1) - \mathbf{X}^0\| \leq 2Bh.$$

Therefore, for $t_0 \leq s \leq t_1 = t_0 + h$ the above bound yields

$$\begin{aligned}\|\mathbf{Y}_1^+(s) - \mathbf{X}(s)\| &\leq \|\mathbf{Y}_1^+(s) - \mathbf{Y}_1^+(t_1)\| + \|\mathbf{Y}_1^+(t_1) - \mathbf{X}(t_1)\| + \|\mathbf{X}(t_1) - \mathbf{X}(s)\| \\ &\leq 4Bh.\end{aligned}$$

Since $\mathbf{Y}_1^-(t_0) = \mathbf{Y}_1^+(t_1)$, we have further

$$\begin{aligned}\|\mathbf{Y}_1^-(s) - \mathbf{X}(s)\| &\leq \|\mathbf{Y}_1^-(s) - \mathbf{Y}_1^-(t_0)\| + \|\mathbf{Y}_1^+(t_1) - \mathbf{X}(t_1)\| + \|\mathbf{X}(t_1) - \mathbf{X}(s)\| \\ &\leq 4Bh.\end{aligned}$$

With regard to $\mathbf{Y}_2^+(t_0) = \mathbf{Y}_1^-(t_1)$, we similarly bound

$$\begin{aligned}\|\mathbf{Y}_2^+(s) - \mathbf{X}(s)\| &\leq \|\mathbf{Y}_2^+(s) - \mathbf{Y}_2^+(t_0)\| + \|\mathbf{Y}_1^-(t_1) - \mathbf{Y}_1^-(t_0)\| \\ &\quad + \|\mathbf{Y}_1^+(t_1) - \mathbf{X}(t_1)\| + \|\mathbf{X}(t_1) - \mathbf{X}(s)\| \\ &\leq 5Bh.\end{aligned}$$

Hence, taking the norm of (2.32) for each perturbation term and using (2.23) we obtain

$$\begin{aligned}\|E_i^\pm\| &= \left\| \mathrm{P}_i^\pm(\mathbf{Y}_i^\pm(t_0)) \int_{t_0}^{t_1} \Delta(s, \mathbf{Y}_i^\pm(s))\, ds \right\| \\ &\leq \int_{t_0}^{t_1} \|\Delta(s, \mathbf{Y}_i^\pm(s))\|\, ds \\ &\leq \int_{t_0}^{t_1} L\|\mathbf{Y}_i^\pm(s) - \mathbf{X}(s)\| + 2\varepsilon\, ds\end{aligned}$$

and so, collecting the above estimates, this results in

$$\|E_1^+\| \leq h\big(4BLh + 2\varepsilon\big), \quad \|E_1^-\| \leq h\big(4BLh + 2\varepsilon\big) \quad \text{and} \quad \|E_2^+\| \leq h\big(5BLh + 2\varepsilon\big).$$

Finally, this gives the stated error bound for the Lie–Trotter splitting scheme. □

We are now in the position to prove the main result of this chapter.

Proof of Theorem 2.4. We first bound the local error $\Phi_F(t_1, t_0, \mathbf{Y}^0) - \mathbf{Y}^1$ by using the triangle inequality, where we use the result of Lemma 2.5 and compare the solutions of

$$\begin{aligned}&\dot{\mathbf{A}}(t) = F(t, \mathbf{A}(t)), && \mathbf{A}(t_0) = \mathbf{Y}^0 \\ \text{and} \quad &\dot{\mathbf{X}}(t) = M(t, \mathbf{X}(t)), && \mathbf{X}(t_0) = \mathbf{X}^0\end{aligned}$$

after one time step, which are denoted by $\Phi_F(t_1, t_0, \mathbf{Y}^0)$ and $\mathbf{X}(t_1)$, respectively. Following condition *(3)* in Assumption 2.3, we have

$$\|M(t, \mathbf{X}(t)) - F(t, \mathbf{X}(t))\| = \|R(t, \mathbf{X}(t))\| \leq \varepsilon$$

and by condition *(1)*, F is Lipschitz continuous. Hence, we obtain by Grönwall's inequality

$$\begin{aligned}\|\Phi_F(t_1,t_0,\mathbf{Y}^0)-\mathbf{X}(t_1)\| &\le e^{L(t_1-t_0)}\|\mathbf{Y}^0-\mathbf{X}^0\| + e^{L(t_1-t_0)}\int_{t_0}^{t_1} e^{-L|s-t_0|}\varepsilon\, ds\\ &\le e^{Lh}\|\mathbf{Y}^0-\mathbf{X}^0\| + e^{Lh}h\varepsilon\\ &= e^{Lh}\big(h(4BLh+2\varepsilon)+h\varepsilon\big),\end{aligned}$$

which together with Lemma 2.5 yields an estimate of the local error that compares the solution of the Lie–Trotter splitting method with the exact solution of problem (2.19) when starting with a low-rank initial value $\mathbf{Y}^0 \in \mathcal{M}$, i.e.,

$$\begin{aligned}\|\Phi_F(t_1,t_0,\mathbf{Y}^0)-\mathbf{Y}^1\| &\le \|\Phi_F(t_1,t_0,\mathbf{Y}^0)-\mathbf{X}(t_1)\| + \|\mathbf{X}(t_1)-\mathbf{Y}^1\|\\ &\le \big(e^{Lh}(4BLh+3\varepsilon)+(9BLh+4\varepsilon)\big)h.\end{aligned}$$

This estimate is visualized in Figure 2.5.

For simplicity of notation, we denote $c = c(B,L,h,\varepsilon) =: e^{Lh}(4BLh+3\varepsilon)+(9BLh+4\varepsilon)$ for the remainder of the proof. Further, the solution operator $\Phi_F(t,s,\cdot)$ of the differential equation with different initial values $\mathbf{A}, \widetilde{\mathbf{A}} \in \mathbb{R}^{n_1\times n_2}$ satisfies, again due to the Lipschitz continuity of F and the Grönwall inequality,

$$\|\Phi_F(t,s,\mathbf{A})-\Phi_F(t,s,\widetilde{\mathbf{A}})\| \le e^{L(t-s)}\|\mathbf{A}-\widetilde{\mathbf{A}}\| \quad \text{for all} \quad \mathbf{A},\widetilde{\mathbf{A}} \in \mathbb{R}^{n_1\times n_2},\ t>s. \tag{2.35}$$

To bound the global error, we propagate the local errors until the final time $T = t_0 + nh$ and then add the transported errors up. This procedure is described as Lady Windermere's fan, see [HNW93]. For an illustration of this procedure, see Figure 2.5.

So let $\mathbf{Y}^k$ be the solution of the Lie–Trotter projector-splitting integrator after k time steps, for $k = 0,\dots,n-1$. The local errors are then given by $\Phi_F(t_{k+1},t_k,\mathbf{Y}^k)-\mathbf{Y}^{k+1}$. Now, propagating those local errors by the flow Φ_F until time step $t_n = t_0 + nh$ yields

$$\begin{aligned}&\|\Phi_F(t_n,t_k,\mathbf{Y}^k)-\Phi_F(t_n,t_{k+1},\mathbf{Y}^{k+1})\|\\ &= \big\|\Phi_F\big(t_n,t_{k+1},\Phi_F(t_{k+1},t_k,\mathbf{Y}^k)\big)-\Phi_F(t_n,t_{k+1},\mathbf{Y}^{k+1})\big\|\\ &\le e^{L(t_n-t_{k+1})}\|\Phi_F(t_{k+1},t_k,\mathbf{Y}^k)-\mathbf{Y}^{k+1}\|\\ &\le (e^{Lh})^{n-k-1}ch.\end{aligned}$$

Now, the accumulation of the propagated local errors results in the global error of the

projector-splitting integrator, which is given by

$$\begin{aligned}
\|\Phi_F(t_n,t_0,\mathbf{Y}^0)-\mathbf{Y}^n\| &\le \|\Phi_F(t_n,t_0,\mathbf{Y}^0)-\Phi_F(t_n,t_1,\mathbf{Y}^1)\| \\
&\quad + \|\Phi_F(t_n,t_1,\mathbf{Y}^1)-\Phi_F(t_n,t_2,\mathbf{Y}^2)\| \\
&\quad + \dots \\
&\quad + \|\Phi_F(t_n,t_{n-2},\mathbf{Y}^{n-2})-\Phi_F(t_n,t_{n-1},\mathbf{Y}^{n-1})\| \\
&\quad + \|\Phi_F(t_n,t_{n-1},\mathbf{Y}^{n-1})-\mathbf{Y}^n)\| \\
&\le ch\cdot\big((e^{Lh})^{n-1}+(e^{Lh})^{n-2}+\dots+(e^{Lh})+1\big) \\
&= ch\cdot\sum_{k=0}^{n-1}(e^{Lh})^k \\
&= ch\Big(\frac{e^{Lnh}-1}{e^{Lh}-1}\Big) \\
&\le ch\frac{e^{Lnh}-1}{Lh} \\
&= c(B,L,h,\varepsilon)\frac{e^{L(T-t_0)}-1}{L} \\
&\le (4+3e^{Lh_0})\frac{e^{L(T-t_0)}-1}{L}\cdot\varepsilon \\
&\quad + (9+4e^{Lh_0})B(e^{L(T-t_0)}-1)\cdot h,
\end{aligned}$$

where h_0 is an upper bound of the stepsize h. Defining

$$c_1 = (4+3e^{Lh_0})\frac{e^{L(T-t_0)}-1}{L} \qquad \text{and} \qquad c_2 = (9+4e^{Lh_0})B(e^{L(T-t_0)}-1) \tag{2.36}$$

gives the main part of the stated error bound. Remarkably, the constants c_1 and c_2 are independent of possibly appearing small singular values in the approximation matrix.

Now, since the stated problem (2.18) starts with $\mathbf{A}(t_0)$, which is not necessarily a matrix of low rank, we also have to estimate the propagated error of the distance between the initial value $\mathbf{A}(t_0)$ and the starting value $\mathbf{Y}^0$. Using assumption *(4)* and the stability of the flow operator (2.35), this yields the propagated initial error at final time, given by

$$\|\Phi_F(t_n,t_0,\mathbf{A}^0)-\Phi_F(t_n,t_0,\mathbf{Y}^0)\| \le e^{L(T-t_0)}\|\mathbf{A}^0-\mathbf{Y}^0\|.$$

Finally, by defining

$$c_0 = e^{L(T-t_0)} \tag{2.37}$$

and collecting the above error bounds, as depicted in Figure 2.4, the global error of the Lie–Trotter splitting scheme at $t_n = t_0 + nh$ is bounded by

$$\begin{aligned}
\|\mathbf{A}(t_n)-\mathbf{Y}^n\| &\le \|\Phi_F(t_n,t_0,\mathbf{A}^0)-\Phi_F(t_n,t_0,\mathbf{Y}^0)\| + \|\Phi_F(t_n,t_0,\mathbf{Y}^0)-\mathbf{Y}^n\| \\
&\le c_0\delta + c_1\varepsilon + c_2 h.
\end{aligned}$$

The Strang splitting scheme is formed by concatenating a half-step of the Lie–Trotter scheme with a half-step of its adjoint. Now, since the same error bound holds for the adjoint scheme, we conclude that it is also valid for the Strang splitting scheme. □

2.4 Error bounds for specific situations

The above error analysis is given for the case when the matrix $\mathbf{A}(t)$ that needs to be approximated is given implicitly by the vector field $F(t, \cdot)$ and in case when the subproblems are solved exactly. In this section, we will handle three further situations, where we lean on [KLW16]. First, we will show in Section 2.4.1 that the error bound simplifies if $\mathbf{A}(t)$ is given explicitly. Second, we will handle the case when the substeps within the integration procedure are solved inexactly, e.g. when a numerical integrator such as a Runge–Kutta method is applied for solving the subproblems, see Section 2.4.2. Third, we discuss in Section 2.4.3, how the error bound improves in case when a one-sided Lipschitz condition is imposed, in order to alleviate the Lipschitz condition on F.

2.4.1 The explicit case

We consider the case where $\mathbf{A}(t)$ is a given time-dependent matrix to which we search an approximation matrix $\mathbf{Y}(t)$ of rank r. In this case the matrix projector-splitting integrator solves

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y}(t))\dot{\mathbf{A}}(t), \qquad \mathbf{Y}(t_0) = \mathbf{Y}^0\,.$$

We have already seen in (2.13) in Section 2.1.2 that we can determine the solutions to the corresponding subproblems in closed form, where due to the independence of the solution in each subproblem, the integrator just uses the increments $\mathbf{A}(t_1) - \mathbf{A}(t_0)$. If we can split up the possibly full rank matrix $\mathbf{A}(t)$ into a low-rank matrix $\mathbf{X}(t) \in \mathcal{M}$ and a perturbation term $\mathbf{R}(t)$ of possibly full rank for all t, i.e.,

$$\mathbf{A}(t) = \mathbf{X}(t) + \mathbf{R}(t),$$

then the derivative of $\mathbf{A}(t)$, which is projected onto $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$, satisfies

$$\dot{\mathbf{A}}(t) = \mathbf{M}(t) + \dot{\mathbf{R}}(t),$$

where, similarly to (2.20), $\mathbf{X}(t)$ solves $\dot{\mathbf{X}}(t) = \mathbf{M}(t)$ with $\mathbf{X}(t_0) = \mathbf{X}^0 \in \mathcal{M}$. In order to be able to apply the error analysis from Section 2.3 to the present situation, we impose

$$\|\mathbf{R}(t_0)\| \leq \delta \qquad \text{and} \qquad \|\dot{\mathbf{R}}(t)\| \leq \varepsilon, \tag{2.38}$$

such that conditions *(3)* and *(4)* in Assumption 2.3 are satisfied. Note that $\mathbf{M}(t) \in \mathcal{T}_{\mathbf{Y}}\mathcal{M}$, since $\mathbf{X}(t)$ is of low rank r. Further, since $F(t, \mathbf{Y}) = \dot{\mathbf{A}}(t)$ is independent of $\mathbf{Y}$, the Lipschitz constant is $L = 0$. Thus, we are in the situation of Theorem 2.4, where the error bound can be adapted appropriately for the present explicit case. We will not provide an error analysis here, just briefly comment on the steps within the proof of Lemma 2.5 and Theorem 2.4, where equations or estimates change due to the explicit case.

In the first step of Lemma 2.5, the intension is to rewrite the differential equation for $\mathbf{Y}(t)$ in terms of $\dot{\mathbf{X}}(t)$. Now, since $F(t, \mathbf{Y}(t))$ is solution independent, we have

$$\begin{aligned}\dot{\mathbf{Y}}(t) &= \mathrm{P}(\mathbf{Y}(t))\dot{\mathbf{A}}(t) = \mathrm{P}(\mathbf{Y}(t))\mathbf{M}(t) + \mathrm{P}(\mathbf{Y}(t))\dot{\mathbf{R}}(t) = \mathbf{M}(t) + \mathrm{P}(\mathbf{Y}(t))\dot{\mathbf{R}}(t) \\ &= \dot{\mathbf{X}}(t) + \mathrm{P}(\mathbf{Y}(t))\dot{\mathbf{R}}(t).\end{aligned} \tag{2.39}$$

Hence, at first glance, it seems that there is no need to insert $\dot{\mathbf{X}}(t) = \mathbf{M}(t)$ as is done in (2.21), since we already have the desired form by means of $\dot{\mathbf{X}}(t)$. But here, we have to be careful concerning the projected remainder term: we cannot bound it by simply taking the norm, since this would involve a dependence on singular values of the projection, which we want to avoid. Therefore, we do insert a zero by adding $\dot{\mathbf{X}}(t)$ and subtracting $\mathbf{M}(t)$ as is done in (2.21), since this enables us to bound the perturbation term $\Delta(t, \mathbf{Y})$ within the modified right-hand side of (2.39):

$$\|\Delta(t, \mathbf{Y}(t))\| = \|\big(\mathbf{I} - \mathrm{P}(\mathbf{Y}(t))\big)\dot{\mathbf{R}}(t, \mathbf{Y}(t)) + \dot{\mathbf{R}}(t, \mathbf{X}(t))\| \leq 2\varepsilon.$$

Steps 2–4 hold for this explicit case without any changes. Instead, step 5 contains bounds for the perturbation terms $E_i^{\pm}$, where $\Delta(t, \mathbf{Y}(t))$ comes into play. There, for our situation we have the estimate

$$\|E_i^{\pm}\| \leq \int_{t_0}^{t_1} \|\Delta(s, \mathbf{Y}_i^{\pm}(s))\| \, ds \leq 2(t_1 - t_0)\varepsilon$$

for each perturbation term. Therefore, the error stated in Lemma 2.5 for the explicit case changes to

$$\|\mathbf{Y}^1 - \mathbf{X}(t_1)\| \leq 4(t_1 - t_0)\varepsilon.$$

Following the lines of the proof of Theorem 2.4 with this result, the local error of the Lie–Trotter projector-splitting integrator in the explicit case is then given as

$$\|\Phi_{\dot{\mathbf{A}}}(t_1, t_0, \mathbf{Y}^0) - \mathbf{Y}^1\| \leq 7(t_1 - t_0)\varepsilon.$$

Then, the accumulation of the propagated local errors until final time t_n and the assumption (2.38) about the initial distance results in the global error of the projector-splitting integrator for the explicit case, which we find as

$$\|\mathbf{A}(t_n) - \mathbf{Y}^n\| \leq \delta + 7(t_n - t_0)\varepsilon, \qquad t_0 \leq t_n = T.$$

Note that the error bound depends on the time interval $[t_0, t_n]$, though it is independent of the time step size h.

2.4.2 Inexact solution within the integration steps

Let us consider the case when the projector-splitting integrator does not compute the exact values $\mathbf{Y}_i^{\pm}(t_1)$, but solves the three substeps inexactly. This occurs for example when applying a numerical method, such as a Runge–Kutta method, for determining solutions within the projector-splitting integrator. Another situation, where we solve substeps approximately will be discussed in Chapter 4, where we present the integrator for Tucker tensors. Independently of the reason, let us suppose that we solve the substeps inexactly and obtain

$$\widehat{\mathbf{Y}}_i^{\pm}(t_1) = \Phi_{F_i^{\pm}}(t_1, t_0, \widehat{\mathbf{Y}}_i^{\pm}(t_0)) + \widehat{\mathbf{E}}_i^{\pm}.$$

In the first step, in fact, instead of $\mathbf{K}(t_1)$ a perturbed value $\mathbf{K}(t_1) + \mathbf{E}_\mathbf{K}(t_1)$ is computed and so we find

$$\begin{aligned}\widehat{\mathbf{Y}}_1^+(t_1) &= \big(\mathbf{K}(t_1) + \mathbf{E}_\mathbf{K}(t_1)\big)\,\mathbf{V}^{0,\top} = \mathbf{K}(t_1)\,\mathbf{V}^{0,\top} + \mathbf{E}_\mathbf{K}(t_1)\,\mathbf{V}^{0,\top} \\ &= \mathbf{Y}_1^+(t_1) + \widehat{\mathbf{E}}_1^+,\end{aligned}$$

and the error satisfies

$$\widehat{\mathbf{E}}_1^+ = \mathbf{E}_\mathbf{K}(t_1)\,\mathbf{V}^{0,\top}\,\mathbf{V}^0\,\mathbf{V}^{0,\top} = \mathrm{P}_1^+(\mathbf{Y}_1^+(t_0))\widehat{\mathbf{E}}_1^+.$$

The latter equation is a natural condition from the way the differential equations for the factors $\mathbf{U}, \mathbf{S}, \mathbf{V}$ of $\mathbf{Y} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top$ are actually solved in the algorithm. For the second substep we compute $\mathbf{S}(t_1) + \mathbf{E}_\mathbf{S}(t_1)$ and obtain the approximate solution

$$\begin{aligned}\widehat{\mathbf{Y}}_1^-(t_1) &= \mathbf{U}^1\big(\mathbf{S}(t_1) + \mathbf{E}_\mathbf{S}(t_1)\big)\,\mathbf{V}^{0,\top} = \mathbf{U}^1\,\mathbf{S}(t_1)\,\mathbf{V}^{0,\top} + \mathbf{U}^1\,\mathbf{E}_\mathbf{S}(t_1)\,\mathbf{V}^{0,\top} \\ &= \mathbf{Y}_1^-(t_1) + \widehat{\mathbf{E}}_1^-,\end{aligned}$$

where

$$\widehat{\mathbf{E}}_1^- = \mathbf{U}^1\,\mathbf{U}^{1,\top}\,\mathbf{U}^1\,\mathbf{E}_\mathbf{S}(t_1)\,\mathbf{V}^{0,\top}\,\mathbf{V}^0\,\mathbf{V}^{0,\top} = \mathrm{P}_1^-(\mathbf{Y}_1^-(t_0))\widehat{\mathbf{E}}_1^-.$$

Finally, in the last step instead of $\mathbf{L}(t_1)^\top$, we compute $\mathbf{L}(t_1)^\top + \mathbf{E}_\mathbf{L}$, such that the solution results in

$$\begin{aligned}\widehat{\mathbf{Y}}_2^+(t_1) &= \mathbf{U}^1\big(\mathbf{L}(t_1)^\top + \mathbf{E}_\mathbf{L}(t_1)\big) = \mathbf{U}^1\,\mathbf{L}(t_1)^\top + \mathbf{U}^1\,\mathbf{E}_\mathbf{L}(t_1) \\ &= \mathbf{Y}_2^+(t_1) + \widehat{\mathbf{E}}_2^+,\end{aligned}$$

where

$$\widehat{\mathbf{E}}_2^+ = \mathbf{U}^1\,\mathbf{U}^{1,\top}\,\mathbf{U}^1\,\mathbf{E}_\mathbf{L}(t_1) = \mathrm{P}_2^+(\mathbf{Y}_2^+(t_0))\widehat{\mathbf{E}}_2^+.$$

So in one full step of the method, instead of $\mathbf{Y}^1$ we actually compute

$$\begin{aligned}\widehat{\mathbf{Y}}^1 &= \Phi_{F_2^+}(t_1, t_0, \Phi_{F_1^-}(t_1, t_0, \Phi_{F_1^+}(t_1, t_0, \mathbf{Y}^0) + \widehat{\mathbf{E}}_1^+) + \widehat{\mathbf{E}}_1^-) + \widehat{\mathbf{E}}_2^+ \\ &= \Phi_{G_2^+}(t_1, t_0, \Phi_{G_1^-}(t_1, t_0, \Phi_{G_1^+}(t_1, t_0, \mathbf{Y}^0) + \mathbf{E}_1^+ + \widehat{\mathbf{E}}_1^+) + \mathbf{E}_1^- + \widehat{\mathbf{E}}_1^-) + \mathbf{E}_2^+ + \widehat{\mathbf{E}}_2^+.\end{aligned}$$

Suppose now that the errors are bounded by

$$\|\widehat{\mathbf{E}}_i^\pm\| \le \eta, \tag{2.40}$$

then the bounds for the total error in each step are given by

$$\begin{aligned}\|\mathbf{E}_1^+ + \widehat{\mathbf{E}}_1^+\| &\le h(4BLh + 2\varepsilon + \eta), \\ \|\mathbf{E}_1^- + \widehat{\mathbf{E}}_1^-\| &\le h(4BLh + 2\varepsilon + \eta), \\ \|\mathbf{E}_2^+ + \widehat{\mathbf{E}}_2^+\| &\le h(5BLh + 2\varepsilon + \eta).\end{aligned}$$

Therefore, we need to adapt the assumption about the initial values in Lemma 2.5 appropriately, such that the error estimate becomes

$$\|\widehat{\mathbf{Y}}^1 - \mathbf{X}(t_1)\| \leq h(9BLh + 4\varepsilon + 2\eta).$$

In this situation the error bound of Theorem 2.4 changes, with the same proof, to

$$\|\mathbf{A}(t_n) - \widehat{\mathbf{Y}}^n\| \leq c_0\delta + c_1\varepsilon + c_2 h + c_3\eta,$$

where c_0, c_1 and c_2 are as before, and

$$c_3 = (2 + e^{Lh_0})(e^{L(T-t_0)} - 1)/L. \tag{2.41}$$

In case when the reason for the inexact solution of the substeps comes from the application of a Runge–Kutta method of order p in order to solve the differential equations for $\mathbf{K}, \mathbf{S}$ and $\mathbf{L}$, the bound (2.40) of the additional errors is of size

$$\eta = \mathcal{O}(h^p).$$

2.4.3 A one-sided Lipschitz condition

We have seen in Theorem 2.4 that the constants depend on the Lipschitz constant L of $F(t, \mathbf{Y})$. Now, in case of a stiff differential equation, L becomes very large, which is a drawback, since then the constants and therefore the error bounds increase. However, we can overcome this difficulty mildly by the one-sided Lipschitz condition [HNW93, Section IV.12], which stays moderate in the stiff case. Suppose that with respect to the Frobenius inner product $\langle \cdot, \cdot \rangle$ we have the one-sided Lipschitz bound

$$\langle F(t, \mathbf{Y}) - F(t, \widetilde{\mathbf{Y}}), \mathbf{Y} - \widetilde{\mathbf{Y}} \rangle \leq \ell \|\mathbf{Y} - \widetilde{\mathbf{Y}}\|^2 \quad \text{for all } \mathbf{Y}, \widetilde{\mathbf{Y}} \in \mathbb{R}^{n_1 \times n_2},$$

with $\ell \leq L$ and possibly $\ell \ll L$. In this case, Lemma 2.5 is left unchangend: the bound of the perturbation term $\Delta(t, \mathbf{Y})$ in (2.23) still depends on the Lipschitz constant L. However, the error propagation in the proof of Theorem 2.4 improves to

$$\|\Phi_F(t, s, \mathbf{A}) - \Phi_F(t, s, \widetilde{\mathbf{A}})\| \leq e^{\ell(t-s)} \|\mathbf{A} - \widetilde{\mathbf{A}}\| \quad \text{for all } \mathbf{A}, \widetilde{\mathbf{A}} \in \mathbb{R}^{n_1 \times n_2},\ t > s,$$

where the factor $e^{L(t-s)}$ from (2.35) is replaced by the smaller factor $e^{\ell(t-s)}$. Following the proof with respect to this adaption, we obtain an error bound of the same form as in Theorem 2.4, but compared to (2.37) and (2.36) with improved constants

$$c_0 = e^{\ell(T-t_0)}, \quad c_1 = (4 + 3e^{\ell h_0})(e^{\ell(T-t_0)} - 1)/\ell, \quad c_2 = (9 + 4e^{\ell h_0})BL(e^{\ell(T-t_0)} - 1)/\ell.$$

Note that the constant c_2 still depends on the Lipschitz constant L, which stems from (2.23). In case of inexact solution, the additional constant c_3 in the error bound (2.41) of the previous section is reduced to

$$c_3 = (2 + e^{\ell h_0})(e^{\ell(T-t_0)} - 1)/\ell.$$

2.5 Numerical experiments

In this section, we will illustrate our theoretical results about the error bounds of the projector-splitting integrator. The numerical examples themselves are taken from [KLW16], but the algorithms are relaunched. In Section 2.5.1, we give a striking example that demonstrates the performance of a standard numerical integrator and the projector-splitting integrator in the presence of small singular values. Afterwards, in Section 2.5.2, we show how two matrices can be added, such that the sum becomes a matrix of low rank. In the numerical example in Section 2.5.3, we compare the Lie–Trotter projector-splitting integrator with the Strang splitting method in terms of small singular values. There, we will corroborate the surprising partial result of Theorem 2.4 that the Strang splitting method converges with order one in the time step size h. The last numerical example, given in Section 2.5.4, deals with a stiff differential equation, where the Lipschitz constant of the right-hand side is large. Due to our error analysis, we would expect the splitting integrator to fail in this situation. To the contrary, we will show that the method performs well also in this case.

2.5.1 The effect of small singular values

We illustrate the favorable behavior of the projector-splitting integrator in the presence of small singular values compared to a standard numerical integration method by applying those two explicit methods in order to solve (2.2). We apply the classical 4th-order Runge–Kutta method to solve the system of differential equations (1.18) for $\mathbf{U}, \mathbf{S}$ and $\mathbf{V}$ within the integration method described in Section 1.2, and the (first-order) Lie–Trotter projector-splitting integrator of [LO14], described in Section 2.1. In this example, $\mathbf{A}(t) \in \mathbb{R}^{100\times 100}$ is given explicitly by constructing two 100×100 skew-symmetric matrices $\mathbf{W}_1, \mathbf{W}_2$ and a diagonal matrix $\mathbf{D}$ of the same dimension with exponentially decreasing diagonal elements $d_j = 2^{-j}$, $j = 1, \dots, 100$. By means of those matrices, we generate

$$\mathbf{A}(t) = e^{t\,\mathbf{W}_1} e^{t}\,\mathbf{D}\left(e^{t\,\mathbf{W}_2}\right)^{\top}$$

on the time interval $0 \le t \le 1$. The singular values of $\mathbf{A}(t)$ are $\sigma_j(t) = e^t d_j$, $j = 1, \dots, 100$.

Figure 2.6 shows the approximation errors that is, the Frobenius norm of the difference between the given matrix $\mathbf{A}(t)$ and the numerical solution $\mathbf{Y}^n = \mathbf{U}^n\,\mathbf{S}^n\,\mathbf{V}^{n,\top}$ of rank r obtained with n steps of stepsize h for $t = nh$, at $t = 1$. The errors versus the stepsize are shown for both methods and for different ranks.

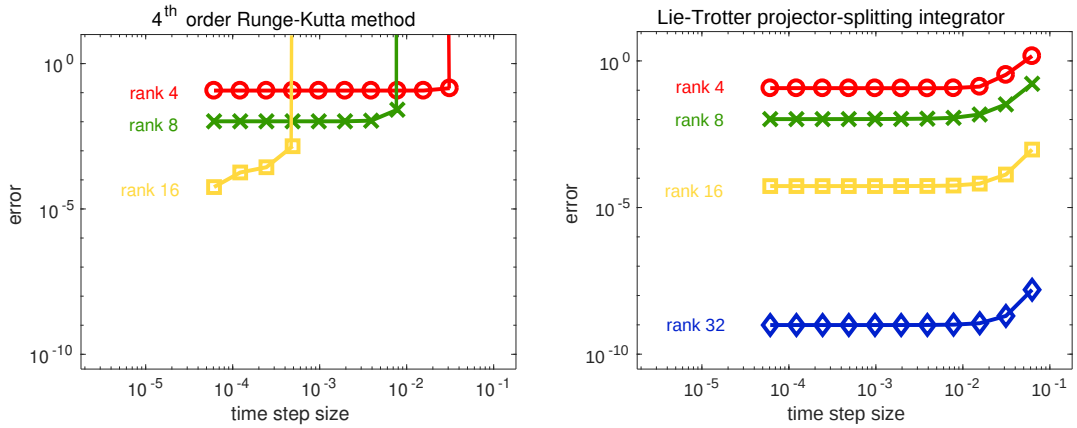

Figure 2.6: Comparing the Runge–Kutta method (left) and the Lie–Trotter integrator (right) for different approximation ranks and stepsizes.

The Runge–Kutta method with approximation ranks $r = \{4, 8, 16\}$ turns out to be stable only for small step sizes and small approximation ranks. For larger approximation ranks r (e.g. $r = 32$) the Runge–Kutta method demands very small step sizes, proportional to σ_r. This restriction is due to the small singular values of $\mathbf{S}(t)$ in (1.18).

In contrast, we are able to choose large time steps for the Lie–Trotter projector-splitting integrator independently of the chosen rank. The error decays linearly with h as $h \to 0$, and decreases with increasing rank, which indicates that the method is accurate and robust with respect to small singular values.

2.5.2 Matrix addition

We consider the addition of two matrices, $\mathbf{C} = \mathbf{A} + \mathbf{B}$, where $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2}$ is a matrix of low rank r, and $\mathbf{B} \in \mathbb{R}^{n_1 \times n_2}$ is an increment. The sum is to be computed approximately with a result in the low-rank manifold $\mathcal{M}$. Such truncated (or retracted) additions are required in iterative methods on low-rank matrix manifolds, in particular in optimization problems; see, e.g., [AO15].

In case when $\mathbf{B} \in \mathcal{T}_{\mathbf{A}}\mathcal{M}$, a standard approach is to first compute $\mathbf{A} + \mathbf{B}$, which is of rank $2r$ and then to project the result onto $\mathcal{M}$ using a truncated SVD, see Section 1.1. Note that the truncated SVD gives a best approximation on the manifold. However, this comes to a computational expensive cost of $\mathcal{O}(N^3)$ operations, where $N = \max\{n_1, n_2\}$.

Alternatively, as proposed in [AO15, LOV15], we can perform approximate addition on the low-rank manifold using the projector-splitting integrator. We then solve

$$\dot{\mathbf{Y}}(t) = \mathrm{P}(\mathbf{Y})\,\mathbf{B}, \qquad \mathbf{Y}(0) = \mathbf{A} \tag{2.42}$$

from $t_0 = 0$ up to $t_1 = 1$ using one step of the Lie–Trotter projector-splitting scheme. As before, $\mathrm{P}(\mathbf{Y})$ denotes the orthogonal projection onto the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$ at $\mathbf{Y} \in \mathcal{M}$. We then get an approximation $\mathbf{Y}^1 \in \mathcal{M}$ for $\mathbf{C} = \mathbf{A} + \mathbf{B}$. Note that we never leave the low-rank manifold $\mathcal{M}$ when using the splitting method. Computing a low-rank approximation

to the sum $\mathbf{C} = \mathbf{A} + \mathbf{B}$ by the splitting method requires $\mathcal{O}(Nr^2)$ operations, which is computationally less expensive when assuming that $r \ll N$.

In our numerical experiment we illustrate this procedure for an example with small singular values. We construct $\mathbf{A} \in \mathbb{R}^{100\times 100}$ of rank $r = 10$ with singular values, which exponentially decrease as $\sigma_j = e^{-j}$, $j = 1, \ldots, 10$. We let $\mathbf{B}$ be a random matrix in $\mathcal{T}_{\mathbf{A}}\mathcal{M}$. We add $\mathbf{C} = \mathbf{A} + \mathbf{B}$ directly to get the full-rank solution, and compare this with solving (2.42) using the Lie–Trotter splitting integrator, which gives us the low-rank solution $\mathbf{Y}^1$. We also compare with the projection $\widetilde{\mathbf{Y}}^1$ obtained by a truncated SVD. This comparison is illustrated in Figure 2.7. We see how the error of the splitting method decays as the norm of the increment $\mathbf{B}$ is reduced. Note also how close the solution given by the splitting method is to the SVD approximation, at reduced computational cost.

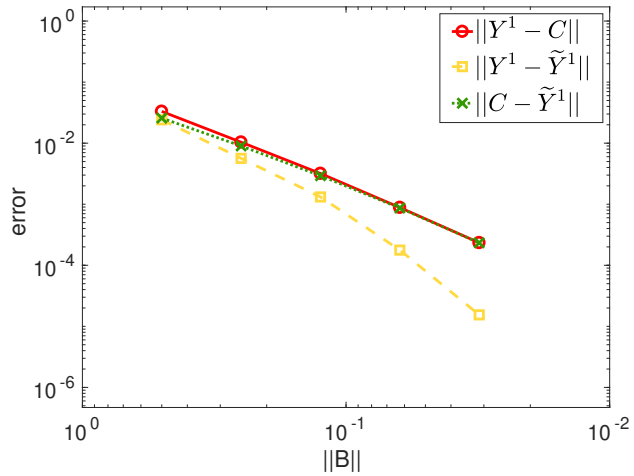

Figure 2.7: Error for matrix addition using the projector-splitting method for tangential increments $\mathbf{B}$ of decreasing norm.

2.5.3 A discrete nonlinear Schrödinger equation for matrices

We consider a discrete nonlinear Schrödinger equation, modeling a Bose–Einstein condensate in an optical lattice [TS01]. The problem reads

$$\begin{aligned} i\dot{\mathbf{A}}(t) &= -\frac{1}{2}\mathbf{L}\mathbf{A}(t) - \frac{1}{2}\mathbf{A}(t)\mathbf{L} - \varepsilon|\mathbf{A}(t)|^2 \odot \mathbf{A}(t), \\ \mathbf{A}_{jk}(0) &= \exp(-(j-j_1)^2/\sigma^2 - (k-k_1)^2/\sigma^2) + \\ &\quad - \exp(-(j-j_2)^2/\sigma^2 - (k-k_2)^2/\sigma^2), \qquad j,k = 1,\ldots,n_1, \end{aligned} \tag{2.43}$$

where $\mathbf{L} = \operatorname{tridiag}(1, 0, 1)$ and $\mathbf{A} \in \mathbb{R}^{n_1\times n_1}$. The squared modulus is taken elementwise and $\odot$ denotes the elementwise product. We use $n_1 = 100$, $\sigma = 10$, $(j_1, k_1) = (60, 50)$, and $(j_2, k_2) = (50, 40)$. Note that $\mathbf{L}$ is *not* a discretized derivative, but a bounded operator modeling the coupling between nodes in the lattice. Since the Frobenius norm of the exact solution is conserved, the right-hand side of (2.43) is bounded and Lipschitz continuous in a neighborhood around the exact solution.

We let $\mathbf{Y}(t)$ denote an approximation to $\mathbf{A}(t)$ on the low-rank manifold $\mathcal{M}$, with rank $r = 10$. The linear terms in (2.43) map onto the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$, while the nonlinear term does not. This makes the dependence of the error on ε explicit. In the table below we study the effect of varying ε and the time step size h. We show the error in Frobenius norm after solving the problem up to $t = 5$ with different ε and h, using the Lie–Trotter projector-splitting scheme. Each subproblem is solved using the 4th-order Runge–Kutta method with time step size $h = 0.001$. The approximate solution is compared to a full rank reference solution, computed with 4th-order Runge–Kutta using the time step size $h = 0.0005$.

$\varepsilon \setminus h$	1	10^{-1}	10^{-2}	10^{-3}
1	9.83e-2	9.73e-2	9.73e-2	9.73e-2
10^{-1}	1.32e-4	8.63e-5	8.63e-5	8.63e-5
10^{-2}	3.13e-6	3.51e-7	3.44e-7	3.44e-7
10^{-3}	2.47e-7	3.44e-9	1.26e-9	1.26e-9
10^{-4}	2.19e-8	2.58e-10	4.09e-11	4.00e-11

Table 2.1: Error in Frobenius norm after solving (2.43) using the Lie–Trotter splitting scheme with different ε and h.

We see how the error decays with ε as predicted. We also see convergence with respect to h in the bottom rows of the table, albeit not of a clear order. The unclear convergence rate with respect to h may be an indication that the error estimate is not quite sharp.

Next, for the same differential equation (2.43) with the same parameters, we apply the Strang projector-splitting integrator, where we first solve the differential equations for $\mathbf{K}$ and $\mathbf{S}$ for half a time step from $t_0 \to t_0 + h/2$, then we solve the ODE for $\mathbf{L}$ for one full time step $t_0 \to t_0 + h$ and afterwards we solve the ODEs for $\mathbf{S}$ and $\mathbf{K}$ from $t_0 + h/2 \to t_1$. Each subproblem is solved by the classical 4th-order Runge–Kutta method with time step size $h = 0.001$. We show the error of the Strang splitting scheme in Frobenius norm in the following table:

$\varepsilon \setminus h$	1	10^{-1}	10^{-2}	10^{-3}
1	9.73e-2	9.73e-2	9.73e-2	9.73e-2
10^{-1}	9.96e-5	8.63e-5	8.63e-5	8.63e-5
10^{-2}	8.14e-7	3.44e-7	3.44e-7	3.44e-7
10^{-3}	8.76e-8	1.37e-9	1.26e-9	1.26e-9
10^{-4}	5.10e-9	1.19e-10	4.00e-11	4.00e-11

Table 2.2: Numerical evidence for linear convergence of the Strang splitting integrator in case when the perturbation term is small.

The results for the Strang splitting scheme look similar to the results for the Lie–Trotter splitting integrator. Surprisingly, for $\varepsilon = 10^{-4}$, we observe *linear* convergence of the Strang splitting integrator. For a larger ε, the influence of the perturbation term might be too strong, such that we do not see a clear convergence rate for $\varepsilon = \{1, 10^{-1}, 10^{-2}, 10^{-3}\}$. When there are no small non-zero singular values, the standard error estimates for splitting methods are valid and the Strang splitting scheme converges towards $\mathbf{Y}(kh)$ at second order in h. In the presence of small singular values, however, this does not seem to be the case, and due to the h-dependence in the bounds of the remainder term $\|\Delta(t, \mathbf{Y})\|$ this is also not promised by our analysis.

2.5.4 A stiff differential equation

Since the error analysis of the projector-splitting integrator given in Section 2.3 relies on both, the boundedness and Lipschitz continuity of F, it does not transfer directly to stiff problems such as spatially discretized partial differential equations. Numerical evidence however suggests that the projector-splitting scheme is robust and accurate also in this case. We consider the time-dependent Schrödinger equation in two dimensions with a harmonic potential,

$$\begin{aligned} iu_t(x,t) &= -\frac{1}{2}\Delta u(x,t) + \frac{1}{2}x^\top \mathbf{A}\, x\, u(x,t), \qquad x \in \mathbb{R}^2,\ t > 0, \\ u(x,0) &= \pi^{-1/2} \exp\Big(\frac{1}{2}x_1^2 + \frac{1}{2}(x_2 - 1)^2\Big), \\ \text{with} \qquad \mathbf{A} &= \begin{pmatrix} 2 & -1 \\ -1 & 3 \end{pmatrix}. \end{aligned}$$

As the right-hand side contains a second order differential operator, its spatial semidiscretization scales as Δx^{-2}, where Δx is the spatial gridsize and therefore the Lipschitz constant of the right-hand side is of the same magnitude. While the initial data is of rank 1, the non-diagonal potential will increase the effective rank of the solution during time evolution. We discretize the problem using Fourier collocation with $m \times m$ grid points on $\Omega = [-7.5, 7.5]^2$. The spatially localized solution is essentially supported within Ω. The approximate solution at the respective grid points is arranged in an $m \times m$ matrix. We

solve low-rank approximations to the problem with ranks $r = 1, 2, \ldots, 20$ using the Lie–Trotter splitting scheme, integrating up to the time $t = 5$. We use $m = 64$ and $m = 128$, and time steps of length $h = 0.02$ and $h = 0.01$. The subproblems are solved to high accuracy by approximating the action of the matrix exponential in a Krylov subspace, see, e.g., [HL97, Saa92], generated by the Arnoldi process [Arn51]. We compare the low-rank approximation to a full-rank reference solution computed by standard Fourier collocation, see, e.g., [Boy01], and Arnoldi time stepping with $m = 128$ and $h = 0.01$. The error is measured in the Frobenius norm, scaled such that it approximates the continuous $L^2(\Omega)$-norm. We depict it in the following figure:

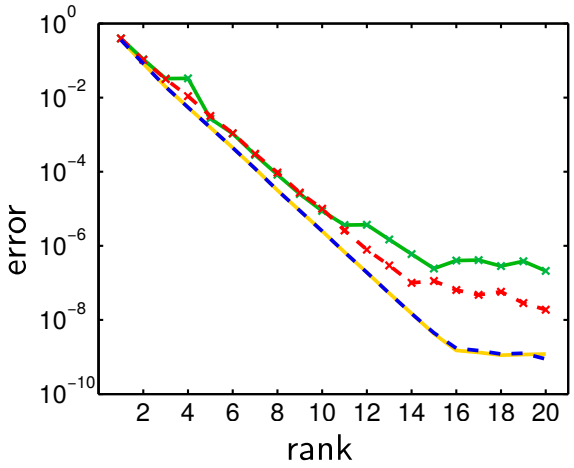

Figure 2.8: Error at different approximation ranks when solving the Schrödinger equation on an $m \times m$ spatial grid. We use $m = 64$ (dashed) and $m = 128$ (solid) grid points per dimension, and the time steps $h = 0.02$ ($\times$) and $h = 0.01$ (plain).

The error decreases exponentially with the rank, which indicates that the method is robust with respect to small singular values also for stiff problems. For the time step $h = 0.02$ we see how the error at high approximation ranks is slightly larger for the finer spatial grid, suggesting a dependence on the Lipschitz constant. The dependence is however mild, and the method much more robust with respect to stiffness than explained by the theory presented in Theorem 2.4.

3 A low-rank splitting integrator for stiff matrix differential equations

We have ended the previous chapter with a numerical example that illustrates a good performance of the projector-splitting integrator also for a stiff matrix differential equation in order to compute low-rank approximations of matrix ODEs. To our best knowledge, it is not known or shown why the projector-splitting integrator has this favorable behavior with regard to stiff differential equations. The difficulty in its error analysis is the usage of the Lipschitz constant of the right-hand side. To omit this difficulty, we will propose an integration method for stiff matrix differential equations, which in contrast can be analyzed without introducing the Lipschitz constant of the right-hand side.

We are dealing with a specific right-hand side of the differential equation, of which the solution needs to be approximated in terms of low rank. The class of stiff and semi-linear matrix differential equations we consider in the time interval $t_0 \leq t \leq T$ is given by

$$\dot{\mathbf{A}}(t) = \mathbf{B}\,\mathbf{A}(t) + \mathbf{A}(t)\,\mathbf{B}^\top + G(t, \mathbf{A}(t)), \qquad \mathbf{A}(t_0) = \mathbf{A}^0,$$

where $\mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_1}$ is the unknown matrix that satisfies the differential equation and $G : [t_0, \infty) \times \mathbb{R}^{n_1 \times n_1} \to \mathbb{R}^{n_1 \times n_1}$ is a given function. We aim to find a low-rank approximation $\mathbf{Y}(t) \in \mathcal{M} = \{\mathbf{Y}(t) \in \mathbb{R}^{n_1 \times n_1} : \operatorname{rank} \mathbf{Y}(t) = r\}$ to the solution of the above matrix differential equation. We propose an integration scheme, which handles the stiff part of the differential equation by simply splitting it off and consider two subproblems, which can be treated separately. For both arising subproblems we compute a low-rank solution by following the dynamical low-rank approximation proposed in [KL07]. The subproblem for the linear stiff part can be integrated explicitly and efficiently by use of exponential integrators. For the nonlinear non-stiff subproblem we apply the projector-splitting integrator proposed in [LO14]. This procedure is derived in Section 3.1, where we also give a practical algorithm that is simple and efficient. Afterwards, we perform an error analysis in Section 3.2 that does not require the Lipschitz constant of the stiff right-hand side of the above matrix differential equation and further shows the key property of the method being robust with respect to small singular values, which might appear in case of over-approximation, i.e., when choosing the approximation rank rather large. The method we propose proves to be of first order. In Section 3.3 we discuss the order of convergence when applying a second order method, such as the Strang splitting scheme. As a special case of the above stiff matrix differential equation, we consider differential Lyapunov equations (DLEs) in Section 3.4, which are of crucial importance in many applications, e.g.,

Kalman filtering [Kal60, KB61, AG15] or model reduction of linear time-varying systems [LSS16, San04]. Another important class of matrix differential equations that follow the structure of the above given stiff matrix differential equation are differential Riccati equations (DREs) [Rei72, BL18]. They play an essential role in optimal and robust control problems [Men07], optimal filtering [DS87] and differential games [Baş91, BM17]. Finally, we illustrate the favorable behavior of the proposed method for stiff matrix ODEs with the help of numerical examples given in Section 3.6.

This chapter is based on a joint work of the author with A. Ostermann and C. Piazzola, see [OPW18].

3.1 The low-rank Lie–Trotter splitting integrator

We consider the following matrix differential equation

$$\dot{\mathbf{A}}(t) = \mathbf{B}\,\mathbf{A}(t) + \mathbf{A}(t)\,\mathbf{B}^\top + G(t, \mathbf{A}(t)), \qquad \mathbf{A}(t_0) = \mathbf{A}^0, \tag{3.1}$$

for $t_0 \leq t \leq T$, where $\mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_1}$ is the unknown solution matrix and the function $G : [t_0, \infty) \times \mathbb{R}^{n_1 \times n_1} \to \mathbb{R}^{n_1 \times n_1}$ is supposed to be nonlinear. The matrix $\mathbf{B} \in \mathbb{R}^{n_1 \times n_1}$ is time-independent. In many cases, where we have to deal with a stiff matrix differential equation of type (3.1), there is a parabolic partial differential equation underlying, where the elliptic operator is assumed to generate a strongly continuous semigroup. After discretizing this parabolic partial differential equation in space, we obtain the matrix $\mathbf{B}$, which is the spatial discretization of the elliptic differential operator. Therefore, the stiffness of (3.1), which is given by the linear part $\mathbf{B}\,\mathbf{A}(t) + \mathbf{A}(t)\,\mathbf{B}^\top$, is induced by the matrix $\mathbf{B}$. The exact full-rank solution of the above differential equation can be represented by the variation-of-constants formula as

$$\mathbf{A}(t) = \mathrm{e}^{(t-t_0)\,\mathbf{B}}\,\mathbf{A}(t_0)\mathrm{e}^{(t-t_0)\,\mathbf{B}^\top} + \int_{t_0}^{t} \mathrm{e}^{(t-s)\,\mathbf{B}} G(s, \mathbf{A}(s))\mathrm{e}^{(t-s)\,\mathbf{B}^\top}\,\mathrm{d}s.$$

We aim to compute an approximate solution $\mathbf{Y}(t) \in \mathcal{M}$ to $\mathbf{A}(t)$, which is of low rank r with $r \ll n_1$. The difficulty is the stiffness of the right-hand side of (3.1), which makes a direct application of the dynamical low-rank approximation by employing the projector-splitting integrator unfeasible. We have seen in the chapter before that in this case, the error bound of the projector-splitting integrator depends on the Lipschitz constant of the full right-hand side. In order to elude difficulties with the Lipschitz constant of stiff matrix differential equations of the class (3.1), the key idea is to separate the ODE into a stiff and a non-stiff subproblem, respectively. For both subproblems we compute a low-rank solution.

3.1.1 Splitting into two subproblems

The idea behind splitting away the stiff part from the non-stiff part of the differential equation (3.1) is to benefit from the independent integration of the two arising subproblems, which are of the form, for $t_0 \leq t \leq T$,

$$\dot{\mathbf{A}}_1(t) = \mathbf{B}\,\mathbf{A}_1(t) + \mathbf{A}_1(t)\,\mathbf{B}^\top, \qquad \mathbf{A}_1(t_0) = \mathbf{A}_1^0 \tag{3.2}$$

and

$$\dot{\mathbf{A}}_2(t) = G(t, \mathbf{A}_2(t)), \qquad \mathbf{A}_2(t_0) = \mathbf{A}_2^0. \tag{3.3}$$

We denote the solutions to the subproblems (3.2) and (3.3) at $t_0 + h$ with initial values $\mathbf{A}_1^0$ and $\mathbf{A}_2^0$ by the solution operators $\Phi_h^{\mathbf{B}}(\mathbf{A}_1^0)$ and $\Phi_h^G(\mathbf{A}_2^0)$, respectively. For an introduction to splitting methods we refer to [HLW06] and [MQ02]. We pursue the strategy to solve the differential equations for $\mathbf{A}_2(t)$ and $\mathbf{A}_1(t)$ subsequently by applying the Lie–Trotter splitting scheme with time step size h, i.e.,

$$\mathcal{L}_h := \Phi_h^{\mathbf{B}} \circ \Phi_h^G. \tag{3.4}$$

We will refer to this scheme as *full-rank Lie–Trotter splitting*.

It results in an approximation $\mathbf{A}^1$ of the solution $\mathbf{A}(t)$ of (3.1) at $t = t_0 + h$. Starting with $\mathbf{A}_2^0 = \mathbf{A}^0$, we obtain

$$\mathbf{A}(t) \approx \mathbf{A}^1 = \mathcal{L}_h\, \mathbf{A}^0 = \Phi_h^{\mathbf{B}} \circ \Phi_h^G(\mathbf{A}^0).$$

The exact solution of the homogeneous problem (3.2) is given by

$$\mathbf{A}_1(t_0 + h) = \mathrm{e}^{h\,\mathbf{B}}\, \mathbf{A}_1^0\, \mathrm{e}^{h\,\mathbf{B}^\top}.$$

To continue with the full-rank Lie–Trotter splitting in time, we take the approximate solution $\mathcal{L}_h\, \mathbf{A}^0$ as initial value and apply the Lie–Trotter splitting scheme in order to solve the differential equations (3.2) and (3.3), respectively, which yields an approximation $\mathbf{A}^2 = \mathcal{L}_h\, (\mathcal{L}_h\, \mathbf{A}^0)$. Proceeding with this approach until the final time $t_n = t_0 + nh = T$, we obtain the approximation

$$\mathbf{A}^n = \mathcal{L}_h^n\, \mathbf{A}^0,$$

which is an approximation to the matrix $\mathbf{A}(t)$ at $t = t_0 + nh$.

Now, $\mathbf{A}_1(t)$ is the result of the action of a matrix exponential and therefore, it can be efficiently computed also for large time step sizes h. Numerical methods of choice are Krylov subspace methods [Saa92, HL97], Taylor interpolation [AMH11] and interpolation at Leja points [CKOR16]. Moreover, efficient implementations on GPUs are possible, see, e.g., [EO13].

The approximate solution $\mathbf{A}^1 = \mathcal{L}_h\, \mathbf{A}^0$ is a *full-rank* matrix approximation to $\mathbf{A}(t_1)$ after one time step. Since we aim to compute a rank r approximation $\mathbf{Y}(t)$ to $\mathbf{A}(t)$ at the time grid points, we next determine low-rank solutions of (3.2) and (3.3).

3.1.2 The low-rank integrator

We first consider the stiff subproblem (3.2). Due to the linearity of the stiff part, we observe that for any $\mathbf{Y} \in \mathcal{M}$, $\mathbf{B}\,\mathbf{Y} + \mathbf{Y}\,\mathbf{B}^\top \in \mathcal{T}_Y\mathcal{M}$ and thus, (3.2) defines a vector field on the low-rank manifold $\mathcal{M}$. Denoting the rank r approximation to $\mathbf{A}_1(t)$ by $\mathbf{Y}_1(t) \in \mathcal{M}$, this can be retraced by a simple calculation, where we apply the orthogonal projection $\mathrm{P}(\mathbf{Y}_1)$

defined in (2.3) in Section 2.1 onto the right-hand side of (3.2). Since $\mathbf{Y}_1 \in \mathcal{M}$, it admits the factorization $\mathbf{Y}_1 = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top$ and we obtain

$$\begin{aligned}
\mathrm{P}(\mathbf{Y}_1)\left(\mathbf{B}\,\mathbf{Y}_1 + \mathbf{Y}_1\,\mathbf{B}^\top\right) &= \left(\mathbf{B}\,\mathbf{Y}_1 + \mathbf{Y}_1\,\mathbf{B}^\top\right)\mathbf{V}\,\mathbf{V}^\top - \mathbf{U}\,\mathbf{U}^\top\left(\mathbf{B}\,\mathbf{Y}_1 + \mathbf{Y}_1\,\mathbf{B}^\top\right)\mathbf{V}\,\mathbf{V}^\top \\
&\quad + \mathbf{U}\,\mathbf{U}^\top\left(\mathbf{B}\,\mathbf{Y}_1 + \mathbf{Y}_1\,\mathbf{B}^\top\right) \\
&= \mathbf{B}\,\mathbf{Y}_1 + \mathbf{Y}_1\,\mathbf{B}^\top\,\mathbf{V}\,\mathbf{V}^\top - \mathbf{U}\,\mathbf{U}^\top\,\mathbf{B}\,\mathbf{Y}_1 - \mathbf{Y}_1\,\mathbf{B}^\top\,\mathbf{V}\,\mathbf{V}^\top \\
&\quad + \mathbf{U}\,\mathbf{U}^\top\,\mathbf{B}\,\mathbf{Y}_1 + \mathbf{Y}_1\,\mathbf{B} \\
&= \mathbf{B}\,\mathbf{Y}_1 + \mathbf{Y}_1\,\mathbf{B}^\top,
\end{aligned}$$

i.e., the stiff part was already in the tangent space of the low-rank manifold $\mathcal{M}$. Hence for an initial value on the low-rank manifold $\mathcal{M}$, the solution of (3.2) stays in $\mathcal{M}$, see also [HM12, Lemma 1.22]. This means that subproblem (3.2) is rank-preserving and so starting with a rank r initial value $\mathbf{Y}_1^0$, the solution of

$$\dot{\mathbf{Y}}_1(t) = \mathbf{B}\,\mathbf{Y}_1(t) + \mathbf{Y}_1(t)\,\mathbf{B}^\top, \qquad \mathbf{Y}_1(t_0) = \mathbf{Y}_1^0 \tag{3.5}$$

stays of rank r for all times. Therefore it is sufficient but crucial to start the integration with a low-rank initial value and apply a suitable time integration method, such as an exponential integrator [HO10, AMH11] in order to obtain a rank r approximation $\mathbf{Y}_1 \approx \mathbf{A}_1$.

Instead, for the second subproblem (3.3) we employ the dynamical low-rank approach [KL07]. There, we project the right-hand side of the nonstiff subproblem (3.3) orthogonally onto the tangent space $\mathcal{T}_{\mathbf{Y}_2(t)}\mathcal{M}$. This results in an evolution equation for $\mathbf{Y}_2(t)$, which is of the form

$$\dot{\mathbf{Y}}_2(t) = \mathrm{P}(\mathbf{Y}_2(t))G(t, \mathbf{Y}_2(t)), \qquad \mathbf{Y}_2(t_0) = \mathbf{Y}_2^0, \tag{3.6}$$

where the initial value $\mathbf{Y}_2^0$ is a rank r matrix and the orthogonal projection is denoted by P. This differential equation needs to be solved numerically. Now, standard integrators show difficulties due to the presence of small singular values, which we have already discussed in Section 1.4. Therefore, we apply the projector-splitting integrator in order to obtain a low-rank approximation to (3.6), which is proven in Section 2.3 to be robust with regard to small singular values. Its integration scheme is described in detail in Section 2.1.1 and the practical integration procedure for the implementation is given in Section 2.1.2. We denote the solution of the projector-splitting integrator applied to (3.6) after one time step of size h by the solution operator φ_h^G. After having applied the projector-splitting integrator to (3.6), the resulting low-rank approximation after one time step at $t_0 + h$ is

$$\mathbf{Y}_2^1 = \varphi_h^G(\mathbf{Y}_2^0).$$

In a nutshell, the integration method we propose consists of first splitting the matrix differential equation (3.1), and then approximating the subproblems (3.2) and (3.3) with respect to low-rank. Hence combining the flow φ_h^G of the low-rank solution of (3.3) with the exact flow $\Phi_h^{\mathbf{B}}$ of (3.2), which is of low rank when starting with a low-rank initial

data, yields the desired approximation matrix $\mathbf{Y}(t)$. We call this procedure the *low-rank Lie–Trotter splitting* and denote it by

$$\mathcal{I}_h := \Phi_h^{\mathbf{B}} \circ \varphi_h^G. \tag{3.7}$$

Therefore, starting with a rank r approximation $\mathbf{Y}^0$ to $\mathbf{A}^0$, we obtain the low-rank approximation of the solution of (3.1) at $t = t_0 + h$, given by

$$\mathbf{Y}(t) \approx \mathbf{Y}^1 = \mathcal{I}_h(\mathbf{Y}^0) = \Phi_h^{\mathbf{B}} \circ \varphi_h^G(\mathbf{Y}^0). \tag{3.8}$$

In order to continue with the low-rank Lie–Trotter splitting in time, we take the solution $\mathcal{I}_h(\mathbf{Y}^0)$ of the first step as the initial value for the subsequent step and again apply the integration scheme. After n time steps, we obtain the low-rank approximation to $\mathbf{Y}(t)$ at $t_n = t_0 + nh = T$, given by

$$\mathbf{Y}^n = \mathcal{I}_h^n(\mathbf{Y}^0).$$

3.1.3 Algorithmic description of the integrator

The implementation of the low-rank Lie–Trotter splitting integrator first follows the lines of the practical algorithm of the projector-splitting integrator given in Section 2.1.2. Afterwards, in order to compute the low-rank approximation of the linear subproblem (3.5), it benefits from the SVD-like factorization of the low-rank initial value, since it simply has to build products. We do not build the intermediate approximation matrix $\mathbf{Y}_2$ after having updated the basis matrices, denoted by $\overline{\mathbf{U}}, \overline{\mathbf{S}}$ and $\overline{\mathbf{V}}^\top$. Instead, we write

$$\mathrm{e}^{h\,\mathbf{B}}\,\mathbf{Y}_2\,\mathrm{e}^{h\,\mathbf{B}^\top} = \Big(\mathrm{e}^{h\,\mathbf{B}}\overline{\mathbf{U}}\Big)\overline{\mathbf{S}}\Big(\overline{\mathbf{V}}^\top \mathrm{e}^{h\,\mathbf{B}^\top}\Big) = \Big(\mathrm{e}^{h\,\mathbf{B}}\overline{\mathbf{U}}\Big)\overline{\mathbf{S}}\Big(\mathrm{e}^{h\,\mathbf{B}}\overline{\mathbf{V}}\Big)^\top$$

and first build the products with the matrix exponentials. Afterwards, we perform two QR decompositions of the resulting matrices, which gives us the updated basis matrices $\mathbf{U}$ and $\mathbf{V}$ and the matrix $\mathbf{S}$, which is build up from a product of three matrices, i.e., we have

$$\Big(\mathrm{e}^{h\,\mathbf{B}}\overline{\mathbf{U}}\Big)\overline{\mathbf{S}}\Big(\mathrm{e}^{h\,\mathbf{B}}\overline{\mathbf{V}}\Big)^\top \overset{\text{QR}}{=} \Big(\mathbf{U}\,\acute{\mathbf{S}}\Big)\overline{\mathbf{S}}\Big(\grave{\mathbf{S}}^\top\,\mathbf{V}^\top\Big) = \mathbf{U}\Big(\acute{\mathbf{S}}\overline{\mathbf{S}}\grave{\mathbf{S}}^\top\Big)\,\mathbf{V}^\top = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top,$$

which we then take as low-rank approximation. Hence, note that solving the linear differential equation (3.5) requires computing the matrix exponential, which can simply be determined by the MATLAB-routine `expm`, or, of course, by a Krylov subspace method for computing the action of a matrix exponential, see [Saa92, HL97]. Afterwards we perform two QR decompositions. Hence, from the computational point of view, we do not solve a differential equation here.

In contrast, the differential equations for $\mathbf{K}, \mathbf{S}$ and $\mathbf{L}$ need to be solved numerically and we therefore apply approximation methods such as a Runge–Kutta method.

The description of the practical integration procedure for the time interval $[t_0, t_n]$ with time steps $(j-1)h = t_{j-1} \to t_j = jh$ for $j = 1, \ldots, n$ is given in Algorithm 3. It computes

factors of a low-rank approximation $\mathbf{Y}^n = \mathbf{U}^n \, \mathbf{S}^n \, \mathbf{V}^{n,\top}$ after n time steps, which is taken as an approximation to $\mathbf{Y}(t_n)$.

Algorithm 3: Low-rank Lie–Trotter splitting integrator

Data: Low-rank matrix $\mathbf{Y}^0 = \mathbf{U}^0 \, \mathbf{S}^0 \, \mathbf{V}^{0,\top}$, $G(t, \mathbf{Y})$, t_0, t_n, h
Result: Approximation matrix $\mathbf{Y}^n = \mathbf{U}^n \, \mathbf{S}^n \, \mathbf{V}^{n,\top}$

begin
  for $j = 1$ **to** n **do**
    set $\mathbf{K}^{j-1} = \mathbf{U}^{j-1} \, \mathbf{S}^{j-1}$
    solve $\dot{\mathbf{K}}(t) = G\big(t, \mathbf{K}(t) \, \mathbf{V}^{j-1,\top}\big) \, \mathbf{V}^{j-1}$,
     with initial value $\mathbf{K}(t_{j-1}) = \mathbf{K}^{j-1}$ and return $\mathbf{K}^j = \mathbf{K}(t_j)$
    compute QR factorization $\mathbf{K}^j = \overline{\mathbf{U}}^j \widehat{\mathbf{S}}^j$
    solve $\dot{\mathbf{S}}(t) = -\overline{\mathbf{U}}^{j,\top} G\big(t, \overline{\mathbf{U}}^j \mathbf{S}(t) \, \mathbf{V}^{j-1,\top}\big) \, \mathbf{V}^{j-1}$,
     with initial value $\mathbf{S}(t_{j-1}) = \widehat{\mathbf{S}}^j$ and return $\widetilde{\mathbf{S}}^{j-1} = \mathbf{S}(t_j)$
    set $\mathbf{L}^{j-1,\top} = \widetilde{\mathbf{S}}^{j-1} \, \mathbf{V}^{j-1,\top}$
    solve $\dot{\mathbf{L}}^{j,\top}(t) = \overline{\mathbf{U}}^{j,\top} G\big(t, \overline{\mathbf{U}}^j \mathbf{L}(t)^\top\big)$,
     with initial value $\mathbf{L}(t_{j-1})^\top = \mathbf{L}^{j-1,\top}$ and return $\mathbf{L}^{j,\top} = \mathbf{L}(t_j)^\top$
    compute QR factorization $\mathbf{L}^j = \overline{\mathbf{V}}^j \overline{\mathbf{S}}^{j,\top}$
    set $\mathbf{U}_{\mathbf{B}}^j = \exp(h \, \mathbf{B}) \overline{\mathbf{U}}^j$
    compute QR factorization $\mathbf{U}_{\mathbf{B}}^j = \mathbf{U}^j \, \acute{\mathbf{S}}^j$
    set $\mathbf{V}_{\mathbf{B}}^j = \exp(h \, \mathbf{B}) \overline{\mathbf{V}}^j$
    compute QR factorization $\mathbf{V}_{\mathbf{B}}^j = \mathbf{V}^j \, \grave{\mathbf{S}}^j$
    set $\mathbf{S}^j = \acute{\mathbf{S}}^j \, \overline{\mathbf{S}}^j \, \grave{\mathbf{S}}^{j,\top}$
  set $\mathbf{Y}^n = \mathbf{U}^n \, \mathbf{S}^n \, \mathbf{V}^{n,\top}$

Note that in most applications, the low-rank initial value $\mathbf{Y}^0$ is not known beforehand. So in order to initialize the algorithm, we first perform a truncated SVD as described in Section 1.1, since we know by the Theorem of Eckart and Young that it results in a best approximation to the given $\mathbf{A}^0$.

3.2 Error analysis of the low-rank Lie–Trotter splitting integrator

In this section we perform an error analysis of the proposed method. The low-rank Lie–Trotter splitting integrator (3.7) is constructed in a way, which enables us to show error bounds without introducing the Lipschitz constant of the full right-hand side of the given differential equation (3.1), nor of the stiff subproblem (3.2). Also, we benefit from the error analysis of the matrix projector-splitting integrator given in Section 2.3 in the sense that the constants of the error bound are not affected by (small) singular values.

Before we start proving the actual error bound, we first describe the framework in which it can be carried out in favor of the organization of the proof. In the following, we are given an initial data $\mathbf{A}^0$ and a final integration time T such that the matrix differential

equation (3.1) has a solution $\mathbf{A}(t)$ for $t_0 \leq t \leq T$. We assume that when starting with a low-rank matrix $\mathbf{Y}^0$, the exact rank r solution

$$\mathbf{Y}(t) = \mathrm{e}^{(t-t_0)\,\mathbf{B}}\,\mathbf{Y}^0\,\mathrm{e}^{(t-t_0)\,\mathbf{B}^\top} + \int_{t_0}^{t} \mathrm{e}^{(t-s)\,\mathbf{B}} \mathrm{P}(\mathbf{Y}(s)) G(s, \mathbf{Y}(s)) \mathrm{e}^{(t-s)\,\mathbf{B}^\top}\,\mathrm{d}s$$

of the matrix differential equation (3.1) exists for $t_0 \leq t \leq T$.

Assumption 3.1. *We assume that*

(1) there exist $\omega \in \mathbb{R}$ and a uniform constant $c > 0$, such that the matrix $\mathbf{B}$ satisfies

$$\|\mathrm{e}^{t\,\mathbf{B}}\,\mathbf{Z}\,\mathrm{e}^{t\,\mathbf{B}^\top}\| \leq \mathrm{e}^{t\omega}\|\mathbf{Z}\|, \tag{3.9}$$

$$\|\mathrm{e}^{t\,\mathbf{B}}(\mathbf{B}\,\mathbf{Z} + \mathbf{Z}\,\mathbf{B}^\top)\mathrm{e}^{t\,\mathbf{B}^\top}\| \leq \frac{1}{t} c \mathrm{e}^{t\omega}\|\mathbf{Z}\|, \tag{3.10}$$

for all $t_0 \leq t \leq T$ and all $\mathbf{Z} \in \mathbb{R}^{n_1 \times n_1}$,

(2) G is continuously differentiable,

(3) G is in the tangent space $\mathcal{T}_{\mathbf{Y}}\mathcal{M}$ up to a small perturbation term:

$$G(t, \mathbf{Y}) = M(t, \mathbf{Y}) + R(t, \mathbf{Y}),$$

where $M(t, \mathbf{Y}) \in \mathcal{T}_{\mathbf{Y}}\mathcal{M}$ and $\|R(t, \mathbf{Y})\| \leq \varepsilon \quad \forall\ \mathbf{Y} \in \mathcal{M}, \quad \forall\ t_0 \leq t \leq T$,

(4) the initial value $\mathbf{A}^0 \in \mathbb{R}^{n_1 \times n_1}$ and the starting value $\mathbf{Y}^0 \in \mathcal{M}$ of the numerical method are δ-close:

$$\|\mathbf{A}^0 - \mathbf{Y}^0\| \leq \delta.$$

We take a closer look on the situation in which we analyze the error behavior of the low-rank Lie–Trotter splitting by discussing the prerequisites.

(1) As announced at the beginning of this chapter, we focus here on the case, where the class of matrix differential equations of the form (3.1) is induced by a spatial discretization of a parabolic partial differential equation. We assume the constant c to be uniform in grid size within the spatial discretization. The elliptic operator of the parabolic partial differential equation is assumed to generate a strongly continuous semigroup in a suitable Banach space. Here, the matrix $\mathbf{B}$ stems from the space discretization of the elliptic operator within the partial differential equation and therefore it is reasonable to assume that it also generates a strongly continuous semigroup. Hence, we conclude that the matrix $\mathbf{B}$ satisfies the exponential boundedness (3.9), see, e.g., [EN06, Proposition 1.4] and the parabolic smoothing property (3.10). These estimates are typically known in case of a vector differential equation, where they have a modified form. For ease of recognition that this assumption is a matrix analogon, we explicate how to transform from the matrix case to the vector case (and vice versa):

Let us define the matrix operator F as

$$F(\mathbf{A}) = \mathbf{B}\,\mathbf{A} + \mathbf{A}\,\mathbf{B}^\top, \quad \mathbf{A} \in \mathbb{R}^{n_1 \times n_1}.$$

Denoting $\operatorname{vec}(\mathbf{B}\,\mathbf{A} + \mathbf{A}\,\mathbf{B}^\top) = \mathcal{B}a$, where $a = \operatorname{vec}(\mathbf{A})$, we transform both sides of the above equation into the vectorized form

$$\operatorname{vec}(F(\mathbf{A})) = \mathcal{B}a.$$

Then, the above bounds can be translated using the vector 2-norm $\|\cdot\|_2$ into

$$\|\mathrm{e}^{t\mathcal{B}} z\|_2 \leq \mathrm{e}^{t\omega}\|z\|_2,$$
$$\|\mathrm{e}^{t\mathcal{B}}\mathcal{B}z\|_2 \leq \frac{1}{t}c\mathrm{e}^{t\omega}\|z\|_2,$$

for all $t > 0$ and all $z = \operatorname{vec}(\mathbf{Z}) \in \mathbb{R}^{n_1^2}$. These properties are well known in the context of semigroup theory for strongly elliptic operators, for further details see, e.g., [Paz83, EN06].

(2) Assuming that the nonlinearity G is continuously differentiable implies the actual properties we require for proving an error estimate:

- G is Lipschitz continuous:

$$\|G(t,\mathbf{Y}) - G(t,\widetilde{\mathbf{Y}})\| \leq L\|\mathbf{Y} - \widetilde{\mathbf{Y}}\|, \qquad \text{for all } \mathbf{Y}, \widetilde{\mathbf{Y}} \in \mathbb{R}^{n_1 \times n_1}$$

- G is bounded:

$$\|G(t,\mathbf{Y})\| \leq B. \tag{3.11}$$

In fact, we only need the weaker assumption of G being continuously differentiable in a neighborhood of the exact solution $\mathbf{A}(t)$ for all $t_0 \leq t \leq T$. This would imply that G is required to be locally Lipschitz continuous and also locally bounded. Later in the proof, we will apply Lady Windermere's fan [HNW93], where we sum up the transported local errors. There, we would have different Lipschitz constants depending on the current approximation matrix in each local error within the sum, which is technically cumbersome to deal with. Hence, it is more convenient to impose a global Lipschitz constant for ease of estimation, but we note that our error estimate still holds under this weaker local condition.

To avoid notational confusion, we point out that in contrast to the bold face $\mathbf{B}$, which denotes a matrix, we denote the bound of G as a plain capital B.

(3) Similarly as in Assumption 2.3 in Chapter 2, we assume that $G(t,\mathbf{Y})$ consists of a tangential part $M(t,\mathbf{Y})$ and a small perturbation term $R(t,\mathbf{Y})$. This means that G, when evaluated along the low-rank solution, is in the tangent space up to a small remainder of size ε. This assumption is crucial in order to have a good low-rank approximation, since if the remainder is large, low-rank approximation is inappropriate.

As a consequence of the condition under consideration and by the boundedness of G, we can estimate the tangential part of G as

$$\|M(t,\mathbf{Y})\| = \|G(t,\mathbf{Y}) - R(t,\mathbf{Y})\| \leq B + \varepsilon.$$

For computational simplicity within estimates in the proof of the upcoming theorem, we choose the bound B in (3.11) to be large enough, such that M has the same bound as G, denoted again by B, i.e.,

$$\|M(t, \mathbf{Y})\| \leq B.$$

Of course, our result also holds if M has a bound that is different from B.

(4) We impose this condition, since we cannot expect a good approximation result, if the rank r initial value $\mathbf{Y}^0$ is far away from the given full-rank initial matrix $\mathbf{A}^0$. In order to guarantee a small distance between those initial values in practice, we perform a truncated SVD factorization of the matrix $\mathbf{A}^0$, since this decomposition results in a best rank r approximation $\mathbf{Y}^0$.

Having discussed the assumptions, we are now in the situation to state the convergence result of the integration method proposed in Section 3.1.

Theorem 3.2. *Under Assumption 3.1, the error of the low-rank Lie–Trotter splitting integrator at $t_n = t_0 + nh$, with step size $h > 0$, is bounded by*

$$\left\|\mathbf{A}(t_0 + nh) - \mathcal{I}_h^n(\mathbf{Y}^0)\right\| \leq c_0\delta + c_1\varepsilon + c_2 h(1 + |\log h|),$$

where c_0, c_1 and c_2 only depend on ω, B, L and T.

In order to facilitate the convergence analysis of the low-rank Lie–Trotter splitting integrator, we study the global error by analyzing the errors that appear due to the construction of the integration scheme and so in total make up the error stated in Theorem 3.2. We collect the contributing terms in the subsequent list, where we follow the chronology of the integration procedure:

(i) First, we split the stiff matrix differential equation (3.1) into a stiff (3.2) and a non-stiff (3.3) subproblem. In the error analysis, we compare the exact full rank solution $\mathbf{A}(t)$ at final time step $T = t_0 + nh$ of the given problem (3.1) with the solution after having splitted this differential equation and then composed the exact solutions of the two arising subproblems. This is the global error of the *full-rank* Lie–Trotter splitting (3.4), which we denote by

$$E_{\text{sp}}^n = \mathbf{A}(t_0 + nh) - (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{A}^0). \tag{3.12}$$

We perform an error analysis in the proof of Proposition 3.3, where we state the error bound.

(ii) After having splitted the differential equation (3.1), we determine the solution of the Lie–Trotter splitting integrator when starting on the one hand with a full-rank initial value $\mathbf{A}^0$ and on the other hand when starting with a low-rank matrix $\mathbf{Y}^0$ for the integration of the two subproblems. Comparing the difference of the solutions depending on the initial data, we find

$$E_\delta^n = (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{A}^0) - (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{Y}^0).$$

In fact, this is the propagation of the δ-difference between the full-rank initial data $\mathbf{A}^0$ and its low-rank approximation $\mathbf{Y}^0$ by the Lie–Trotter splitting method.

We analyze this error in the proof of Theorem 3.2.

(iii) Finally, within the proposed integrator, we determine a low-rank solution of the two subproblems by applying the low-rank Lie–Trotter splitting, where we use the projector-splitting integrator amongst others. We compare this low-rank approximation $\mathcal{I}_h^n$ with the solution of the full-rank Lie–Trotter splitting integrator $\mathcal{L}_h$ with low-rank initial value $\mathbf{Y}^0$, i.e.,

$$E_{\mathrm{lr}}^n = (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{Y}^0) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^n(\mathbf{Y}^0).$$

A proof of the distance between the full-rank and the low-rank splitting integration procedure, respectively, is performed in Proposition 3.4.

For a better overview of the appearing errors, we depict the composition of the global error of the low-rank Lie–Trotter splitting method in the following figure.

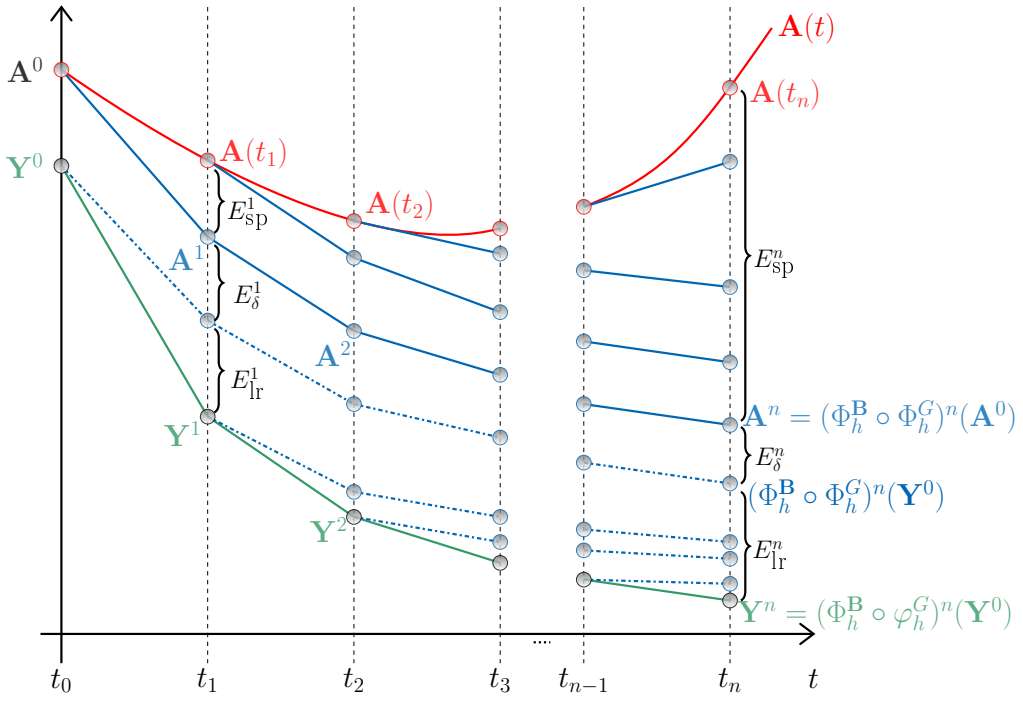

Figure 3.1: Schematic illustration of the convergence analysis. The uppermost curve (in red) depicts the exact solution $\mathbf{A}(t)$ of (3.1), whereas the lowermost (in green) shows the solution obtained by the low-rank Lie–Trotter splitting (3.7). All other lines (in blue) represent the solutions obtained by the application of the Lie–Trotter splitting (3.4) either to a full-rank initial data (solid lines) or to a low-rank initial data (dashed-dotted lines).

The global error in Theorem 3.2 is a composition of three terms, which we study in the following.

We start with the error estimates of the full-rank Lie–Trotter splitting scheme (3.4). The ideas in the proof can be traced back to, e.g., [JL00] and [EO15].

Proposition 3.3. *Under Assumption 3.1, the full-rank Lie–Trotter splitting* (3.4) *is first-order convergent, i.e., its error at $t_n = t_0 + nh$, with time step size $h > 0$, is bounded by*

$$\|\mathbf{A}(t_0 + nh) - (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{A}^0)\| \leq C_3 h(1 + |\log h|).$$

The constant C_3 only depends on ω, B, L and T.

Proof. The solution of the given matrix differential equation (3.1) can be expressed by means of the variation-of-constants formula. Given the initial value $\mathbf{A}(t_{k-1})$, the solution at time $t_k = t_{k-1} + h$ after one time step with step size $h > 0$ is

$$\mathbf{A}(t_k) = \mathrm{e}^{h\,\mathbf{B}}\,\mathbf{A}(t_{k-1})\mathrm{e}^{h\,\mathbf{B}^\top} + \int_0^h \mathrm{e}^{(h-s)\,\mathbf{B}} G\big(t_{k-1}+s, \mathbf{A}(t_{k-1}+s)\big)\mathrm{e}^{(h-s)\,\mathbf{B}^\top}\,\mathrm{d}s, \tag{3.13}$$

for all $k = 1, \ldots, n$. The exact solution $\Phi_h^{\mathbf{B}}$ of the first full-rank subproblem (3.2) for $\mathbf{A}_1(t)$ at t_k with initial value $\mathbf{A}_1(t_{k-1}) = \mathbf{A}_2$ can again be determined by the variation-of-constants formula and is given by

$$\mathbf{A}_1(t_k) = \Phi_h^{\mathbf{B}}(\mathbf{A}_2) = \mathrm{e}^{h\,\mathbf{B}}\,\mathbf{A}_2\,\mathrm{e}^{h\,\mathbf{B}^\top}.$$

The exact solution of the second full-rank subproblem (3.3) for $\mathbf{A}_2(t)$ with initial value $\mathbf{A}_2(t_{k-1})$ can be expressed by applying Taylor expansion, see, e.g. [Tay17, Kön03], with integral form of the remainder, i.e.,

$$\begin{aligned}
\mathbf{A}_2(t_k) &= \Phi_h^G\big((\mathbf{A}_2(t_{k-1})\big) \\
&= \mathbf{A}_2(t_{k-1}) + \dot{\mathbf{A}}_2(t_{k-1})(t_k - t_{k-1}) + \int_{t_{k-1}}^{t_k} (t_k - \tau)\ddot{\mathbf{A}}_2(t_{k-1}+\tau)\,\mathrm{d}\tau \\
&= \mathbf{A}_2(t_{k-1}) + hG(t_{k-1}, \mathbf{A}_2(t_{k-1})) + \int_0^h (h-s)\ddot{\mathbf{A}}_2(t_{k-1}+s)\,\mathrm{d}s,
\end{aligned} \tag{3.14}$$

where we have substituted $\tau = s + t_{k-1}$ in the last equation. Following the Lie–Trotter splitting scheme, we compose those exact solutions $\Phi_h^{\mathbf{B}}$ and Φ_h^G of the two subroblems and obtain the solution of the full-rank Lie–Trotter splitting method, which is given by

$$\begin{aligned}
\mathcal{L}_h\,\mathbf{A}_2(t_{k-1}) &= \Phi_h^{\mathbf{B}} \circ \Phi_h^G(\mathbf{A}_2(t_{k-1})) \\
&= \Phi_h^{\mathbf{B}}\Big(\mathbf{A}_2(t_{k-1}) + hG\big(t_{k-1}, \mathbf{A}_2(t_{k-1})\big) + \int_0^h (h-s)\ddot{\mathbf{A}}_2(t_{k-1}+s)\,\mathrm{d}s\Big) \\
&= \mathrm{e}^{h\,\mathbf{B}}\,\mathbf{A}_2(t_{k-1})\mathrm{e}^{h\,\mathbf{B}^\top} + h\mathrm{e}^{h\,\mathbf{B}} G\big(t_{k-1}, \mathbf{A}_2(t_{k-1})\big)\mathrm{e}^{h\,\mathbf{B}^\top} \\
&\quad + \int_0^h (h-s)\mathrm{e}^{h\,\mathbf{B}}\ddot{\mathbf{A}}_2(t_{k-1}+s)\mathrm{e}^{h\,\mathbf{B}^\top}\,\mathrm{d}s.
\end{aligned}$$

Since the order of integration within the splitting method (3.4) suggests to first solving the second subproblem, the initial value for the full scheme is $\mathbf{A}_2(t_{k-1}) = \mathbf{A}(t_{k-1})$. Then,

with the exact solution (3.13) of the given ODE (3.1), the local error of the Lie–Trotter splitting method at $t_k = t_{k-1} + h$ with time step size $h > 0$ is

$$\begin{aligned} e_{\mathrm{sp}}^k &= \mathbf{A}(t_k) - \mathcal{L}_h\, \mathbf{A}(t_{k-1}) \\ &= \int_0^h \mathrm{e}^{(h-s)\,\mathbf{B}} G\big(t_{k-1}+s, \mathbf{A}(t_{k-1}+s)\big)\mathrm{e}^{(h-s)\,\mathbf{B}^\top}\,\mathrm{d}s \\ &\quad - h\mathrm{e}^{h\,\mathbf{B}} G\big((t_{k-1}, \mathbf{A}(t_{k-1})\big)\mathrm{e}^{h\,\mathbf{B}^\top} - \int_0^h (h-s)\mathrm{e}^{h\,\mathbf{B}}\ddot{\mathbf{A}}_2(t_{k-1}+s)\mathrm{e}^{h\,\mathbf{B}^\top}\,\mathrm{d}s. \end{aligned}$$

In the following, we will rewrite the local error in a simplified form. To this end, let us denote the integrand of the first integral above as

$$f(s) = \mathrm{e}^{(h-s)\,\mathbf{B}} G(t_{k-1}+s, \mathbf{A}(t_{k-1}+s))\mathrm{e}^{(h-s)\,\mathbf{B}^\top},$$

such that

$$\int_0^h f(s)\,\mathrm{d}s = \int_0^h f(0) + f(s) - f(0)\,\mathrm{d}s = \int_0^h \left(f(0) + \int_0^s \dot{f}(\tau)\,\mathrm{d}\tau\right)\,\mathrm{d}s. \tag{3.15}$$

For ease of presentation, we will drop the arguments of the nonlinearity G in the following. In cases where it differs from the one above, we will denote it explicitly.

Due to the fact that a matrix commutes with its exponential, the time derivative of f is given as

$$\begin{aligned} \dot{f}(\tau) &= \frac{\mathrm{d}}{\mathrm{d}\tau}\Big(\mathrm{e}^{(h-\tau)\,\mathbf{B}} G\big(t_{k-1}+\tau, \mathbf{A}(t_{k-1}+\tau)\big)\mathrm{e}^{(h-\tau)\,\mathbf{B}^\top}\Big) \\ &= -\mathbf{B}\,\mathrm{e}^{(h-\tau)\,\mathbf{B}} G\big(t_{k-1}+\tau, \mathbf{A}(t_{k-1}+\tau)\big)\mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \\ &\quad + \mathrm{e}^{(h-\tau)\,\mathbf{B}}\left(\frac{\mathrm{d}}{\mathrm{d}\tau}G\right)\mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \\ &\quad - \mathrm{e}^{(h-\tau)\,\mathbf{B}} G\big(t_{k-1}+\tau, \mathbf{A}(t_{k-1}+\tau)\big)\,\mathbf{B}^\top\,\mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \\ &= -\mathrm{e}^{(h-\tau)\,\mathbf{B}}\Big[\mathbf{B}\,G\big(t_{k-1}+\tau, \mathbf{A}(t_{k-1}+\tau)\big) \\ &\qquad + G\big(t_{k-1}+\tau, \mathbf{A}(t_{k-1}+\tau)\big)\,\mathbf{B}^\top - \frac{\mathrm{d}}{\mathrm{d}\tau}G\Big]\,\mathrm{e}^{(h-\tau)\,\mathbf{B}^\top}. \end{aligned} \tag{3.16}$$

Now, by Assumption 3.1*(2)*, the nonlinearity G is continuously differentiable and hence its time derivative $\ddot{\mathbf{A}}_2(t)$ is bounded. Therefore, when inserting the simplified form (3.15), with (3.16), in the local error, we are left with

$$\begin{aligned} e_{\mathrm{sp}}^k &= \int_0^h f(0)\,\mathrm{d}s + \int_0^h\int_0^s \dot{f}(\tau)\,\mathrm{d}\tau\,\mathrm{d}s \\ &\quad - h\mathrm{e}^{h\,\mathbf{B}} G\big((t_{k-1}, \mathbf{A}(t_{k-1})\big)\mathrm{e}^{h\,\mathbf{B}^\top} - \int_0^h (h-s)\mathrm{e}^{h\,\mathbf{B}}\ddot{\mathbf{A}}_2(t_{k-1}+s)\mathrm{e}^{h\,\mathbf{B}^\top}\,\mathrm{d}s \\ &= -\int_0^h\int_0^s \mathrm{e}^{(h-\tau)\,\mathbf{B}}\Big(\mathbf{B}\,G + G\,\mathbf{B}^\top\Big)\,\mathrm{e}^{(h-\tau)\,\mathbf{B}^\top}\,\mathrm{d}\tau\,\mathrm{d}s \\ &\quad + \int_0^h\int_0^s \mathrm{e}^{(h-\tau)\,\mathbf{B}}\left(\frac{\mathrm{d}}{\mathrm{d}\tau}G\right)\mathrm{e}^{(h-\tau)\,\mathbf{B}^\top}\,\mathrm{d}\tau\,\mathrm{d}s - \int_0^h (h-s)\mathrm{e}^{h\,\mathbf{B}}\ddot{\mathbf{A}}_2(t_{k-1}+s)\mathrm{e}^{h\,\mathbf{B}^\top}\,\mathrm{d}s. \end{aligned} \tag{3.17}$$

For determining an error bound for the full-rank Lie–Trotter splitting method, we bound each term of the local error separately, but we cannot bound the first integral directly, since that would give us an undesired dependence on the Lipschitz constant of the linear part, which is induced by the Frobenius norm of the matrix $\mathbf{B}$ that causes the stiffness. Nevertheless, before we strike a new path in order to bound the first double integral, we first estimate the two remaining terms. We observe that they mainly consist of matrix exponentials and time derivatives of G.

So since by Assumption 3.1 *(1)* and *(2)* the matrix exponential and the nonlinearity G are bounded, the second term in (3.17) can be estimated by

$$\begin{aligned}
\left\| \int_0^h \int_0^s \mathrm{e}^{(h-\tau)\,\mathbf{B}} \left(\frac{\mathrm{d}}{\mathrm{d}\tau} G \right) \mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \,\mathrm{d}\tau \,\mathrm{d}s \right\| &\leq \int_0^h \int_0^s \mathrm{e}^{(h-\tau)\omega} \left\| \frac{\mathrm{d}}{\mathrm{d}\tau} G \right\| \,\mathrm{d}\tau \,\mathrm{d}s \\
&\leq C \mathrm{e}^{h\omega} \left(\frac{1}{\omega^2} \mathrm{e}^{-h\omega} + \frac{h}{\omega} - \frac{1}{\omega^2} \right) \\
&= C(\omega) \left(1 + (h\omega - 1) \mathrm{e}^{h\omega} \right) \\
&= C(\omega) \Big(1 + h\omega + (h\omega)^2 + \frac{1}{2}(h\omega)^3 + \mathcal{O}\big((h\omega)^4\big) \\
&\qquad -1 - h\omega - \frac{1}{2}(h\omega)^2 - \frac{1}{6}(h\omega)^3 - \mathcal{O}\big((h\omega)^4\big) \Big) \\
&\leq C(\omega) h^2,
\end{aligned}$$

where we have used the power series of the exponential function. Note that the constant depends on ω.

With the same arguments, we similarly bound the third term of the local error (3.17), which yields

$$\begin{aligned}
\left\| \int_0^h (h-s) \mathrm{e}^{h\,\mathbf{B}} \ddot{\mathbf{A}}_2(t_{k-1}+s) \mathrm{e}^{h\,\mathbf{B}^\top} \,\mathrm{d}s \right\| &\leq \int_0^h (h-s) \left\| \mathrm{e}^{h\,\mathbf{B}} \ddot{\mathbf{A}}_2(t_{k-1}+s) \mathrm{e}^{h\,\mathbf{B}^\top} \right\| \,\mathrm{d}s \\
&\leq C \mathrm{e}^{h\omega} \frac{h^2}{2} \\
&\leq C \mathrm{e}^{(T-t_0)\omega} \frac{h^2}{2} \\
&\leq C(T,\omega) h^2.
\end{aligned}$$

Therefore, since the last two terms of the local error are proportional to $\mathcal{O}(h^2)$, the local error is reduced to

$$e_{\mathrm{sp}}^k = -\int_0^h \int_0^s \mathrm{e}^{(h-\tau)\,\mathbf{B}} \left(\mathbf{B}\,G + G\,\mathbf{B}^\top \right) \mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \,\mathrm{d}\tau \,\mathrm{d}s + \mathcal{O}(h^2). \tag{3.18}$$

By assumption *(1)*, we have the parabolic smoothing property available, which we apply

in order to bound this local error. Substituting $\alpha = h - \tau$ and $\mathrm{d}\alpha = -\,\mathrm{d}\tau$, it yields

$$\begin{aligned}
\left\| e_{\mathrm{sp}}^{k} \right\| &= \left\| \int_0^h \int_0^s \mathrm{e}^{(h-\tau)\,\mathbf{B}} \left(\mathbf{B}\,G + G\,\mathbf{B}^\top \right) \mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \,\mathrm{d}\tau\,\mathrm{d}s \right\| + \mathcal{O}(h^2) \\
&\leq \int_0^h \int_0^s C \frac{1}{h-\tau} \mathrm{e}^{(h-\tau)\omega} \|G\| \,\mathrm{d}\tau\,\mathrm{d}s \\
&\leq C(B)\mathrm{e}^{h\omega} \int_0^h \int_{h-s}^h \frac{1}{\alpha} \,\mathrm{d}\alpha\,\mathrm{d}s \\
&\leq C(B)\mathrm{e}^{(T-t_0)\omega} \left(h(2|\log(h)| - 1) \right) \\
&\leq C(B,T,\omega) \left(h(2|\log(h)| + 2) \right) \\
&= C(B,T,\omega) h(|\log(h)| + 1).
\end{aligned} \tag{3.19}$$

Now, when determining the global error bound from this local bound, we would lose one order of the time step size h, which would not lead to a promising error analysis, where the expected global error is of order one. Therefore, we purse another approach, where we exploit a recursive form of the global error.

We can retrace the recursive construction by means of the fan in Figure 3.1. The fan for our situation is depicted by the blue solid lines. There, we see that the global error E_{sp}^n at time $T = t_n$ consists of the distance between the propagation of the exact solution $\mathbf{A}(t_{n-1})$ and the numerical solution $\mathbf{A}^{n-1}$ by $\mathcal{L}_h$ plus the local error e_{sp}^n at time t_n, i.e.,

$$E_{\mathrm{sp}}^n = \mathcal{L}_h\,\mathbf{A}(t_{n-1}) - \mathcal{L}_h\,\mathbf{A}^{n-1} + e_{\mathrm{sp}}^n. \tag{3.20}$$

To clarify that this in fact is the global error of the splitting scheme when starting with $\mathbf{A}^0 = \mathbf{A}(t_0)$, we remind that the numerical solution of the Lie–Trotter splitting integrator after $(n-1)$ time steps can be recursively written as $\mathbf{A}^{n-1} = \mathcal{L}_h^{n-1}\,\mathbf{A}^0$. We observe that the local error (3.18) is contained in the global error for $k = n$. Later on, we will benefit from the reduced representation (3.18) within the global error.

In the following, we estimate the global error E_{sp}^n by means of Lady Windermere's fan [HNW93, II.3]. To this end, we first exploit the form of the exact solution (3.14) of the nonlinear subproblem, but with initial values $\mathbf{A}(t_{n-1})$ and $\mathbf{A}^{n-1}$, respectively. Denoting the time derivative of $G(\cdot, \mathbf{A}(t_{n-1}))$ by $\ddot{\mathbf{A}}(\cdot)$ and the time derivative of $G(\cdot, \mathbf{A}^{n-1})$ by $\ddot{\widetilde{\mathbf{A}}}(\cdot)$,

we write

$$\begin{aligned}
\mathcal{L}_h\,\mathbf{A}(t_{n-1}) - \mathcal{L}_h\,\mathbf{A}^{n-1} &= \mathrm{e}^{h\,\mathbf{B}}\big(\Phi_h^G(\mathbf{A}(t_{n-1})) - \Phi_h^G(\mathbf{A}^{n-1})\big)\mathrm{e}^{h\,\mathbf{B}^\top} \\
&= \mathrm{e}^{h\,\mathbf{B}}\big(\mathbf{A}(t_{n-1}) - \mathbf{A}^{n-1}\big)\mathrm{e}^{h\,\mathbf{B}^\top} \\
&\quad + \mathrm{e}^{h\,\mathbf{B}}\Big[h\big(G(t_{n-1},\mathbf{A}(t_{n-1})) - G(t_{n-1},\mathbf{A}^{n-1})\big) \\
&\qquad + \int_0^h (h-s)\left(\ddot{\mathbf{A}}_2(t_{n-1}+s) - \ddot{\widetilde{\mathbf{A}}}_2(t_{n-1}+s)\right)\mathrm{d}s\Big]\,\mathrm{e}^{h\,\mathbf{B}^\top} \\
&= \mathrm{e}^{h\,\mathbf{B}}\big(\mathbf{A}(t_{n-1}) - \mathbf{A}^{n-1}\big)\mathrm{e}^{h\,\mathbf{B}^\top} + \mathrm{e}^{h\,\mathbf{B}}\left[H\big(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\big)\right]\mathrm{e}^{h\,\mathbf{B}^\top} \\
&= \mathrm{e}^{h\,\mathbf{B}}\big(\mathbf{A}(t_{n-1}) - \mathcal{L}_h^{n-1}\,\mathbf{A}^0\big)\mathrm{e}^{h\,\mathbf{B}^\top} + \mathrm{e}^{h\,\mathbf{B}}\left[H\big(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\big)\right]\mathrm{e}^{h\,\mathbf{B}^\top} \\
&= \mathrm{e}^{h\,\mathbf{B}}\big(\mathbf{A}(t_{n-1}) - \big(\Phi_h^{\mathbf{B}}\circ\Phi_h^G\big)^{n-1}\,\mathbf{A}^0\big)\mathrm{e}^{h\,\mathbf{B}^\top} \\
&\quad + \mathrm{e}^{h\,\mathbf{B}}\left[H\big(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\big)\right]\mathrm{e}^{h\,\mathbf{B}^\top} \\
&= \mathrm{e}^{h\,\mathbf{B}}E_{\mathrm{sp}}^{n-1}\mathrm{e}^{h\,\mathbf{B}^\top} + \mathrm{e}^{h\,\mathbf{B}}\left[H\left(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\right)\right]\mathrm{e}^{h\,\mathbf{B}^\top},
\end{aligned}$$

where we denote

$$\begin{aligned}
H\left(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\right) &= h\big(G(t_{n-1},\mathbf{A}(t_{n-1})) - G(t_{n-1},\mathbf{A}^{n-1})\big) \\
&\quad + \int_0^h (h-s)\left(\ddot{\mathbf{A}}_2(t_{n-1}+s) - \ddot{\widetilde{\mathbf{A}}}_2(t_{n-1}+s)\right)\mathrm{d}s,
\end{aligned}$$

and where we have used the form (3.12) for the global splitting error E_{sp}^{n-1}. With this representation of the propagated exact and the propagated numerical solution, where the propagation is executed by $\mathcal{L}_h$, which contains the global error until the previous time step, we observe that, in fact, the global error E_{sp}^n admits a recursive description. Using this recursiveness, we rewrite the global error as

$$\begin{aligned}
E_{\mathrm{sp}}^n &= \mathcal{L}_h\,\mathbf{A}(t_{n-1}) - \mathcal{L}_h\,\mathbf{A}^{n-1} + e_{\mathrm{sp}}^n \\
&= \mathrm{e}^{h\,\mathbf{B}}E_{\mathrm{sp}}^{n-1}\mathrm{e}^{h\,\mathbf{B}^\top} + \mathrm{e}^{h\,\mathbf{B}}H\big(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\big)\mathrm{e}^{h\,\mathbf{B}^\top} + e_{\mathrm{sp}}^n \\
&= \mathrm{e}^{h\,\mathbf{B}}\left(\mathcal{L}_h\,\mathbf{A}(t_{n-1}) - \mathcal{L}_h\,\mathbf{A}^{n-1} + e_{\mathrm{sp}}^{n-1}\right)\mathrm{e}^{h\,\mathbf{B}^\top} + \mathrm{e}^{h\,\mathbf{B}}H\big(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\big)\mathrm{e}^{h\,\mathbf{B}^\top} + e_{\mathrm{sp}}^n \\
&= \mathrm{e}^{h\,\mathbf{B}}\left(\mathrm{e}^{h\,\mathbf{B}}\left(E_{\mathrm{sp}}^{n-2} + H\big(\mathbf{A}(t_{n-2}),\mathbf{A}^{n-2}\big)\right)\mathrm{e}^{h\,\mathbf{B}^\top} + e_{\mathrm{sp}}^{n-1}\right)\mathrm{e}^{h\,\mathbf{B}^\top} \\
&\quad + \mathrm{e}^{h\,\mathbf{B}}H\big(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\big)\mathrm{e}^{h\,\mathbf{B}^\top} + e_{\mathrm{sp}}^n \\
&= \mathrm{e}^{2h\,\mathbf{B}}E_{\mathrm{sp}}^{n-2}\mathrm{e}^{2h\,\mathbf{B}^\top} \\
&\quad + \mathrm{e}^{2h\,\mathbf{B}}H\big(\mathbf{A}(t_{n-2}),\mathbf{A}^{n-2}\big)\mathrm{e}^{2h\,\mathbf{B}^\top} + \mathrm{e}^{h\,\mathbf{B}}H\big(\mathbf{A}(t_{n-1}),\mathbf{A}^{n-1}\big)\mathrm{e}^{h\,\mathbf{B}^\top} \\
&\quad + \mathrm{e}^{h\,\mathbf{B}}e_{\mathrm{sp}}^{n-1}\mathrm{e}^{h\,\mathbf{B}^\top} + e_{\mathrm{sp}}^n \\
&\;\;\vdots \\
&= \mathrm{e}^{nh\,\mathbf{B}}E_{\mathrm{sp}}^0\mathrm{e}^{nh\,\mathbf{B}^\top} + \sum_{k=0}^{n-1}\mathrm{e}^{(n-k)h\,\mathbf{B}}H\big(\mathbf{A}(t_k),\mathbf{A}^k\big)\mathrm{e}^{(n-k)h\,\mathbf{B}^\top} + \sum_{k=1}^{n}\mathrm{e}^{(n-k)h\,\mathbf{B}}e_{\mathrm{sp}}^k\mathrm{e}^{(n-k)h\,\mathbf{B}^\top}.
\end{aligned}$$

We observe that due to the equality of the initial values $\mathbf{A}(t_0) = \mathbf{A}^0$, the first term within the above representation of the global error vanishes, since $E_{\mathrm{sp}}^0 = \mathbf{A}(t_0) - \mathbf{A}^0$ is the null

matrix. In order to bound the second term, we have to bound the matrix $H\big(\mathbf{A}(t_k), \mathbf{A}^k\big)$, which mainly consists of the nonlinear term G and its continuous time derivative $\ddot{\mathbf{A}}_2$. Now, by Assumption 3.1 *(2)*, G is Lipschitz continuous and hence, for all $k = 0, \dots, n-1$, we find

$$\begin{aligned}
\|H(\mathbf{A}(t_k), \mathbf{A}^k)\| &\leq h\|G(t_k, \mathbf{A}(t_k)) - G(t_k, \mathbf{A}^k)\| \\
&\quad + \int_0^h (h-s)\|\ddot{\mathbf{A}}_2(t_k+s) - \tilde{\ddot{\mathbf{A}}}_2(t_k+s)\| \,\mathrm{d}s \\
&\leq hL\|\mathbf{A}(t_k) - \mathbf{A}^k\| + C\frac{h^2}{2} \\
&= C(L)(h\|E_{\mathrm{sp}}^k\| + h^2),
\end{aligned}$$

where we remind that $\mathbf{A}^k$ is the numerical solution of the full-rank Lie–Trotter splitting integrator. Note that the constant C depends on L, the Lipschitz constant of G. Now, using Assumption 3.1 *(1)*, the matrix exponential multiplied by a matrix is bounded and therefore we find for the second term in the above representation of the global error the bound

$$\begin{aligned}
\left\| \sum_{k=0}^{n-1} \mathrm{e}^{(n-k)h\,\mathbf{B}} H\big(\mathbf{A}(t_k), \mathbf{A}^k\big) \mathrm{e}^{(n-k)h\,\mathbf{B}^\top} \right\| &\leq \sum_{k=0}^{n-1} \mathrm{e}^{(n-k)h\omega} \left\| H\big(\mathbf{A}(t_k), \mathbf{A}^k\big) \right\| \\
&\leq \sum_{k=0}^{n-1} \mathrm{e}^{nh\omega} C(L) \left(h\|E_{\mathrm{sp}}^k\| + h^2 \right) \\
&= C(L)\mathrm{e}^{nh\omega} h \left(\sum_{k=0}^{n-1} \|E_{\mathrm{sp}}^k\| + nh \right) \\
&= C(L)\mathrm{e}^{(T-t_0)\omega} h \left(\sum_{k=0}^{n-1} \|E_{\mathrm{sp}}^k\| + (T-t_0) \right) \\
&\leq C(L, T, \omega) h \left(\sum_{k=0}^{n-1} \|E_{\mathrm{sp}}^k\| + 1 \right).
\end{aligned} \tag{3.21}$$

We continue bounding the third term in the representation of the global error. There, we use the simplified form (3.17) of the local error and also Assumption 3.1 *(1)*, such that we obtain

$$
\begin{aligned}
&\left\| \sum_{k=1}^{n} \mathrm{e}^{(n-k)h\,\mathbf{B}} e_{\mathrm{sp}}^{k} \mathrm{e}^{(n-k)h\,\mathbf{B}^\top} \right\| \\
&\leq \sum_{k=1}^{n-1} \left\| \mathrm{e}^{(n-k)h\,\mathbf{B}} e_{\mathrm{sp}}^{k} \mathrm{e}^{(n-k)h\,\mathbf{B}^\top} \right\| + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&= \sum_{k=1}^{n-1} \left\| \mathrm{e}^{(n-k)h\,\mathbf{B}} \left(- \int_0^h \int_0^s \mathrm{e}^{(h-\tau)\,\mathbf{B}} \left(\mathbf{B}\,G + G\,\mathbf{B}^\top \right) \mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \,\mathrm{d}\tau \,\mathrm{d}s + \mathcal{O}(h^2) \right) \mathrm{e}^{(n-k)h\,\mathbf{B}^\top} \right\| \\
&\quad + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&= \sum_{k=1}^{n-1} \int_0^h \int_0^s \left\| \mathrm{e}^{(h-\tau)\,\mathbf{B}} \left(\mathrm{e}^{(n-k)h\,\mathbf{B}} (\mathbf{B}\,G + G\,\mathbf{B}^\top) \mathrm{e}^{(n-k)h\,\mathbf{B}^\top} \right) \mathrm{e}^{(h-\tau)\,\mathbf{B}^\top} \,\mathrm{d}\tau \,\mathrm{d}s \right\| \\
&\quad + (n-1) C h^2 \mathrm{e}^{(n-k)h\omega} + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&\leq \sum_{k=1}^{n-1} \int_0^h \int_0^s \mathrm{e}^{(h-\tau)\omega} \frac{C}{h(n-k)} \mathrm{e}^{(n-k)h\omega} \|G\| \,\mathrm{d}\tau \,\mathrm{d}s + (n-1) C h^2 \mathrm{e}^{(n-k)h\omega} + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&\leq C(B) \mathrm{e}^{nh\omega} \sum_{k=1}^{n-1} \frac{1}{hk} \mathrm{e}^{h\omega} \int_0^h \int_0^s \mathrm{e}^{-\tau\omega} \,\mathrm{d}\tau \,\mathrm{d}s + C n h^2 \mathrm{e}^{nh\omega} + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&= C(B) \mathrm{e}^{(T-t_0)\omega} \frac{1}{h} \sum_{k=1}^{n-1} \frac{1}{k} C(\omega) h^2 + C(T-t_0) h \mathrm{e}^{(T-t_0)\omega} + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&= C(B,T,\omega) \sum_{k=1}^{n-1} \frac{1}{k} C(\omega) h + C(T,\omega) h + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&\leq C(B,T,\omega) h |\log(n)| + C(T,\omega) h + \left\| e_{\mathrm{sp}}^{n} \right\| \\
&\leq C(B,T,\omega) h |\log(h)| + C(T,\omega) h + \left\| e_{\mathrm{sp}}^{n} \right\|,
\end{aligned}
$$

where, in the second to last step, we have approximated the n-th partial sum of the harmonic series by the logarithmic function. We are left with bounding the last local error e_{sp}^{n} of the splitting scheme. Here, we remind that the bound (3.19) of the local error is given for all $k = 1, \ldots, n$ and therefore we can apply it here as well.

In total, we have

$$
\begin{aligned}
\|E_{\mathrm{sp}}^{n}\| &\leq C(L,T,\omega) h \left(\sum_{k=0}^{n-1} \|E_{\mathrm{sp}}^{k}\| + 1 \right) + C(B,T,\omega) h (|\log(h)| + 1) \\
&= C(B,T,\omega) h (1 + |\log(h)|) + \sum_{k=0}^{n-1} C(L,T,\omega) h \|E_{\mathrm{sp}}^{k}\|.
\end{aligned}
$$

The global error bound follows now from a discrete Grönwall inequality, i.e.,

$$
\begin{aligned}
\|E_{\mathrm{sp}}^{n}\| &\leq C_3(B,L,T,\omega) h (1 + |\log(h)|) \mathrm{e}^{nC(L,T,\omega)h} \\
&= C_3(B,L,T,\omega) h (1 + |\log(h)|).
\end{aligned}
$$

□

Having estimated the error of the Lie–Trotter splitting for the given full-rank problem (3.1), we now turn to the low-rank solution of the Lie–Trotter splitting integrator and compare those two methods. This means that we analyze the difference between them, i.e.,

$$E_{\mathrm{lr}}^n = (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{Y}^0) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^n(\mathbf{Y}^0).$$

Although starting with a low-rank initial matrix $\mathbf{Y}^0$, the full-rank Lie–Trotter splitting integrator $\Phi_h^{\mathbf{B}} \circ \Phi_h^G$ does not result in a low-rank matrix. This can be seen from the differential equations (3.3) and (3.2), which are solved consecutively: the ODE for G starts with a low-rank initial value $\mathbf{A}_2(t_0) = \mathbf{Y}^0$, though it results in a full-rank solution, since the right-hand side of this ODE is nonlinear, i.e., it is not in the (linear) tangent space of the low-rank manifold $\mathcal{M}$. Since we solve the first subproblem directly, then, starting with a full-rank initial matrix, the result stays of full rank after one time step. In contrast, the low-rank Lie–Trotter integrator $\Phi_h^{\mathbf{B}} \circ \varphi_h^G$ is characterized by solving a differential equation for the nonlinearity G, which is obtained by the orthogonal projection $\mathrm{P}(\mathbf{Y}_2(t))$ onto $\mathcal{T}_{\mathbf{Y}_2(t)}\mathcal{M}$. Applying the projector-splitting integrator onto this problem on the tangent space $\mathcal{T}_{\mathbf{Y}_2(t)}\mathcal{M}$ yields a low-rank solution, which due to the splitting scheme in turn is the initial value for the linear subproblem (3.5) that is solved directly. Hence, starting with a rank r matrix, the solution of (3.5) will stay of rank r, such that the low-rank Lie–Trotter splitting integrator $\Phi_h^{\mathbf{B}} \circ \varphi_h^G$ results in a low-rank approximation matrix. We analyze next the distance between the full-rank and the low-rank integrator, respectively.

Proposition 3.4. *Under Assumption 3.1, the difference of the full-rank Lie–Trotter and the low-rank Lie–Trotter splitting method at $t_n = t_0 + nh$, with step size $h > 0$, is bounded by*

$$\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{Y}^0) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^n(\mathbf{Y}^0)\| \leq C_1\varepsilon + C_2 h,$$

where the constants C_1 and C_2 only depend on ω, B, L and T.

Note that, from the technical perspective, the bound in this proposition is not an error bound, since we do not compare the result of a numerical method with an exact solution, but we compare solutions of two numerical methods with each other.

The idea for proving this estimate is to follow the strategy of Lady Windermere's fan [HNW93, II.3]: we first consider the appropriate local distances between the full-rank and the low-rank solution in each step and then we propagate them by the stable solution operator until the final time $T = t_0 + nh$, where we eventually add the propagated local distances up and obtain the desired bound. This strategy is visualized by the blue dashed-dotted lines in Figure 3.1.

Lemma 3.5. *Under Assumption 3.1, the local bound of the distance between the full-rank Lie–Trotter and the low-rank Lie–Trotter splitting method at $t_k = t_{k-1} + h$, with step size $h > 0$, is given by*

$$\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)(\mathbf{Y}^{k-1}) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)(\mathbf{Y}^{k-1})\| \leq b_1\varepsilon h + b_2 h^2,$$

where the constants b_1 and b_2 only depend on ω, B, L and T.

Proof. By Assumption 3.1 *(1)*, we employ the exponential boundedness and observe that

$$\begin{aligned}\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)(\mathbf{Y}^{k-1}) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)(\mathbf{Y}^{k-1})\| &= \|\Phi_h^{\mathbf{B}} \circ (\Phi_h^G - \varphi_h^G)(\mathbf{Y}^{k-1})\| \\ &\leq \mathrm{e}^{h\omega}\|(\Phi_h^G - \varphi_h^G)(\mathbf{Y}^{k-1})\|. \end{aligned} \tag{3.22}$$

This time, the difference (3.22) of the two solution operators for the second subproblem (3.3) in fact is a local error, since it compares the exact solution Φ_h^G with the numerical solution φ_h^G. Since φ_h^G is the solution obtained by applying the projector-splitting integrator onto the projected right-hand side, we mainly refer to the error analysis for the robustness result given in Section 2.3 or in [KLW16].

By Assumption 3.1 *(3)*, we can split up the nonlinearity $G(t, \mathbf{Y}_2(t))$ into a linear and a nonlinear part, viz.,

$$G(t, \mathbf{Y}_2(t)) = M(t, \mathbf{Y}_2(t)) + R(t, \mathbf{Y}_2(t)).$$

Now, suppose that there is no nonlinear part, i.e., $R = 0$. Then, the differential equation

$$\dot{\mathbf{W}}(t) = M(t, \mathbf{W}(t)), \qquad \mathbf{W}(t_{k-1}) = \mathbf{W}^{k-1} \tag{3.23}$$

is available with low-rank solution $\mathbf{W}(t_k) \in \mathcal{M}$. The idea of this proof is to relate the exact solution Φ_h^G and the numerical solution φ_h^G, respectively, to the exact solution $\mathbf{W}(t)$ of this auxiliary problem. We interpret both, the full-rank and the low-rank differential equations (3.3) and (3.6), viz.,

$$\dot{\mathbf{A}}_2(t) = G(t, \mathbf{A}_2(t)), \qquad \mathbf{A}_2(t_0) = \mathbf{A}_2^0$$

and

$$\dot{\mathbf{Y}}_2(t) = \mathrm{P}(\mathbf{Y}_2(t))G(t, \mathbf{Y}_2(t)), \qquad \mathbf{Y}_2(t_0) = \mathbf{Y}_2^0,$$

respectively, as differential equations for $\mathbf{W}(t)$ with a perturbation term. For instance, we write the ODE for $\mathbf{Y}_2(t)$ in terms of $\mathbf{W}(t)$ and find

$$\dot{\mathbf{Y}}_2(t) = \dot{\mathbf{W}}(t) + \Delta(t, \mathbf{Y}_2(t)),$$

where $\Delta(t, \mathbf{Y}_2(t)) = G(t, \mathbf{Y}_2(t)) - G(t, \mathbf{W}(t)) - \big(\mathbf{I} - \mathrm{P}(\mathbf{Y}_2(t))\big)R(t, \mathbf{Y}_2(t)) + R(t, \mathbf{W}(t))$.

We need to analyze the influence of the perturbation term in both differential equations onto their solutions, when determined exactly (Φ_h^G) as well as when applying the projector-splitting integrator (φ_h^G), i.e., we are interested in bounds for $\Phi_h^G(\mathbf{Y}^{k-1}) - \mathbf{W}(t_k)$ and $\varphi_h^G(\mathbf{Y}^{k-1}) - \mathbf{W}(t_k)$, for all $k = 1, \ldots, n$

A rigorous error analysis that studies the influence of the perturbation term $\Delta(t, \mathbf{Y}_2(t))$ is given in Lemma 2.5 in Section 2.3. Our Assumption 3.1 fulfills the assumption of that lemma and so we can directly apply its result. Suppose that the distance of the initial values is bounded by $\|\mathbf{Y}^{k-1} - \mathbf{W}^{k-1}\| \leq h(4BLh + 2\varepsilon)$, see Figure 2.4 in Section 2.3 for $\mathbf{X}^{k-1} = \mathbf{W}^{k-1}$, then by following the error estimate of Lemma 2.5, we obtain

$$\|\varphi_h^G(\mathbf{Y}^{k-1}) - \mathbf{W}(t_k)\| \leq h(9BLh + 4\varepsilon). \tag{3.24}$$

Now, for comparing the exact solution of the second full-rank subproblem (3.3) with the exact solution of the unperturbed, auxiliary problem (3.23), we first observe that by Assumption 3.1 *(3)* we have the bound

$$\|M(t, \mathbf{W}(t)) - G(t, \mathbf{W}(t))\| = \|R(t, \mathbf{W}(t))\| \leq \varepsilon.$$

By condition *(2)*, we know that G is Lipschitz continuous and therefore we can apply Grönwall's inequality, which then gives

$$\begin{aligned}\|\Phi_h^G(\mathbf{Y}^{k-1}) - \mathbf{W}(t_k)\| &\leq \mathrm{e}^{L(t_k - t_{k-1})}\|\mathbf{Y}^{k-1} - \mathbf{W}^{k-1}\| + \mathrm{e}^{L(t_k - t_{k-1})}\int_{t_{k-1}}^{t_k} \mathrm{e}^{-L|s - t_{k-1}|}\varepsilon \,\mathrm{d}s \\ &\leq \mathrm{e}^{Lh}\|\mathbf{Y}^{k-1} - \mathbf{W}^{k-1}\| + \mathrm{e}^{Lh}h\varepsilon \\ &\leq \mathrm{e}^{Lh}h(4BLh + 3\varepsilon).\end{aligned}$$

Together with (3.24), this in total yields the local bound

$$\begin{aligned}&\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)(\mathbf{Y}^{k-1}) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)(\mathbf{Y}^{k-1})\| \\ &\leq \mathrm{e}^{h\omega}\left(\|\Phi_h^G(\mathbf{Y}^{k-1}) - \mathbf{W}(t_k)\| + \|\varphi_h^G(\mathbf{Y}^{k-1}) - \mathbf{W}(t_k)\|\right) \\ &\leq \mathrm{e}^{h\omega}\left(h(9BLH + 4\varepsilon) + \mathrm{e}^{Lh}h(4BLh + 3\varepsilon)\right) \\ &\leq \mathrm{e}^{(T-t_0)\omega}\left(\left(4\varepsilon + 3\mathrm{e}^{L(T-t_0)}\varepsilon\right)h + \left(9BL + \mathrm{e}^{L(T-t_0)}4BL\right)h^2\right),\end{aligned}$$

which results in the stated local bound with

$$b_1 = \mathrm{e}^{(T-t_0)\omega}\left(4 + 3\mathrm{e}^{L(T-t_0)}\right) \qquad \text{and} \qquad b_2 = \mathrm{e}^{(T-t_0)\omega}\left(9BL + \mathrm{e}^{L(T-t_0)}4BL\right).$$

□

With this local estimate at hand, we are now in the situation to prove the global bound for $E_{\mathrm{lr}}^n = (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{Y}^0) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^n(\mathbf{Y}^0)$ that compares the solution of the full-rank Lie–Trotter splitting with the solution of the low-rank Lie–Trotter method after n time steps at $T = t_0 + nh$, starting from a low-rank initial value.

Proof of Proposition 3.4. By condition *(2)* of Assumption 3.1, G is Lipschitz continuous and hence we conclude by Grönwall's inequality for $\widetilde{\mathbf{A}}, \widehat{\mathbf{A}} \in \mathbb{R}^{n_1 \times n_1}$ that the solution operator $\mathcal{L}_h = \Phi_h^{\mathbf{B}} \circ \Phi_h^G$ satisfies

$$\begin{aligned}\|\big(\Phi_h^{\mathbf{B}} \circ \Phi_h^G\big)(\widehat{\mathbf{A}}) - \big(\Phi_h^{\mathbf{B}} \circ \Phi_h^G\big)(\widetilde{\mathbf{A}})\| &= \left\|\Phi_h^{\mathbf{B}}\left(\Phi_h^G(\widehat{\mathbf{A}}) - \Phi_h^G(\widetilde{\mathbf{A}})\right)\right\| \\ &\leq \mathrm{e}^{h\omega}\mathrm{e}^{Lh}\|\widehat{\mathbf{A}} - \widetilde{\mathbf{A}}\| \\ &= \mathrm{e}^{(L+\omega)h}\|\widehat{\mathbf{A}} - \widetilde{\mathbf{A}}\|,\end{aligned} \tag{3.25}$$

which shows stability of the full-rank Lie–Trotter splitting method $\mathcal{L}_h$. We will use this stability in order to propagate the local bound from Lemma 3.5 until final time $T = t_0 + nh$ and then accumulate the transported errors. Again, this procedure follows Lady Windermere's fan argument, see Figure 3.1.

We start with propagating the local bound given in Lemma 3.5 by $\mathcal{L}_h$ and find

$$\begin{aligned}
&\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n-k}(\mathbf{Y}^k) - (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n-(k+1)}(\mathbf{Y}^{k+1})\| \\
&= \|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n-(k+1)} \Big((\Phi_h^{\mathbf{B}} \circ \Phi_h^G)(\mathbf{Y}^k)\Big) - (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n-(k+1)}(\mathbf{Y}^{k+1})\| \\
&\leq \mathrm{e}^{(L+\omega)h(t_n - t_{k+1})}\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)(\mathbf{Y}^k) - \mathbf{Y}^{k+1}\| \\
&= \mathrm{e}^{(L+\omega)h(t_n - t_{k+1})}\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)(\mathbf{Y}^k) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)(\mathbf{Y}^k)\| \\
&\leq \mathrm{e}^{(L+\omega)h(n-(k+1))}(b_1\varepsilon h + b_2 h^2).
\end{aligned}$$

Now, we obtain a global bound for E_{lr}^n by adding those propagated local bounds and find

$$\begin{aligned}
&\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n}(\mathbf{Y}^0) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^{n}(\mathbf{Y}^0)\| \\
&\leq \|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n}(\mathbf{Y}^0) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^{n-1}(\mathbf{Y}^1)\| \\
&\quad + \|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n-1}(\mathbf{Y}^1) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^{n-2}(\mathbf{Y}^2)\| \\
&\quad \vdots \\
&\quad + \|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)(\mathbf{Y}^{n-1}) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^{n}(\mathbf{Y}^0)\| \\
&= \sum_{k=0}^{n-1} \|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^{n-k}(\mathbf{Y}^k) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^{n-(k+1)}(\mathbf{Y}^{k+1})\| \\
&\leq (b_1\varepsilon + b_2 h)h \sum_{k=0}^{n-1} \mathrm{e}^{(L+\omega)h(n-(k+1))} \\
&= (b_1\varepsilon + b_2 h)h \sum_{k=0}^{n-1} \mathrm{e}^{(L+\omega)hk} \\
&= (b_1\varepsilon + b_2 h)h \frac{\mathrm{e}^{(L+\omega)nh} - 1}{\mathrm{e}^{(L+\omega)h} - 1} \\
&\leq (b_1\varepsilon + b_2 h)h \frac{\mathrm{e}^{(L+\omega)nh} - 1}{(L+\omega)h} \\
&= (4 + 3\mathrm{e}^{L(T-t_0)})\mathrm{e}^{(T-t_0)\omega} \frac{\mathrm{e}^{(L+\omega)(T-t_0)} - 1}{(L+\omega)}\ \varepsilon \\
&\quad + (9BL + \mathrm{e}^{L(T-t_0)}4BL)\mathrm{e}^{(T-t_0)\omega} \frac{\mathrm{e}^{(L+\omega)(T-t_0)} - 1}{(L+\omega)}\ h,
\end{aligned}$$

which by defining

$$C_1 = (4 + 3\mathrm{e}^{L(T-t_0)})\mathrm{e}^{(T-t_0)\omega} \frac{\mathrm{e}^{(L+\omega)(T-t_0)} - 1}{(L+\omega)}$$

$$\text{and} \quad C_2 = (9BL + \mathrm{e}^{L(T-t_0)}4BL)\mathrm{e}^{(T-t_0)\omega} \frac{\mathrm{e}^{(L+\omega)(T-t_0)} - 1}{(L+\omega)}$$

yields the stated global bound. □

Now, the given problem (3.1) starts with a full-rank initial matrix $\mathbf{A}^0$, but the proposed low-rank Lie–Trotter integrator takes a rank r approximation $\mathbf{Y}^0$ to $\mathbf{A}^0$ as initial value. Therefore, we have to propagate the initial distance $\|\mathbf{A}^0 - \mathbf{Y}^0\| \leq \delta$, see condition *(4)* of

our main assumption. This yields the error term E_δ^n, which we need in order to complete the list of errors that contribute to the global error of the low-rank Lie–Trotter splitting.

Proof of Theorem 3.2. It is sufficient to give a bound for the propagated initial error $\|\mathbf{A}^0 - \mathbf{Y}^0\|$, since bounds for the other contributing terms are given in Proposition 3.3 and Proposition 3.4.

In the proof of Proposition 3.4, we have seen in (3.25) that the solution operator $(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)$ is stable. Therefore, by Grönwall's inequality we find the propagated initial error at final time $T = t_0 + nh$, i.e.,

$$\|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{A}^0) - \left(\Phi_h^{\mathbf{B}} \circ \Phi_h^G\right)^n(\mathbf{Y}^0)\| \leq \mathrm{e}^{(L+\omega)(T-t_0)}\|\mathbf{A}^0 - \mathbf{Y}^0\|.$$

Finally, defining

$$c_0 = \mathrm{e}^{(L+\omega)(T-t_0)}$$

and collecting the error of the full-rank Lie–Trotter splitting shown in Proposition 3.3, the bound of the low-rank Lie–Trotter splitting proven in Proposition 3.4 and the above propagated initial distance, the global error of the low-rank Lie–Trotter splitting integrator is bounded by

$$\|\mathbf{A}(t_0 + nh) - \mathbf{Y}^n\| \leq c_0\delta + c_1\varepsilon + c_2(1 + |\log(h)|),$$

where $c_1 = C_1$ from Proposition 3.4 and c_2 contains the constants C_2 from the same proposition as well as the constant C_3 from Proposition 3.3. □

We point out that in contrast to standard numerical integrators, the low-rank Lie–Trotter splitting integrator is not sensitive to the presence of small singular values. It is constructed in a way, such that it both does not suffer from possibly appearing singular values, and is completely independent of them, whether they are small or large. The low-rank solution of the linear problem (3.6) is computed directly by an exponential integrator, where small singular values do not cause difficulties. Further, they can also occur in the approximation matrix of the nonlinear subproblem. But since we are applying the projector-splitting integrator, which in Section 2.3 is proven to be robust with respect to small singular values, our integration method inherits this favorable property.

3.3 Discussion about the low-rank Strang splitting

The low-rank Lie–Trotter splitting integrator is a first order method. If we want to increase the order of the method, we follow the Strang splitting scheme, see [Str68], which has classical order two in the non-stiff case. There, we split the given differential equation (3.1) in the same way as described in Section 3.1.1 into the subproblems (3.2) and (3.3), respectively. The approximate solution to $\mathbf{A}(t)$ is then given by

$$\mathbf{A}(t) \approx \left(\Phi_{h/2}^{\mathbf{B}} \circ \Phi_h^G \circ \Phi_{h/2}^{\mathbf{B}}\right)(\mathbf{A}^0).$$

We call this integration method the *full-rank Strang splitting*. In order to determine a low-rank solution to the given differential equation (3.1), we then follow the strategy described in Section 3.1.2, where we bring the splitted subproblems by orthogonal projection onto the tangent space of the low-rank manifold and obtain the differential equations for $\mathbf{Y}_1(t)$ and $\mathbf{Y}_2(t)$ given in (3.5) and (3.6), respectively. The differential equation for the stiff and linear part is solved directly, since the right-hand side already defines a vector field. The ODE for the projected nonlinearity G, instead, is solved by the projector-splitting integrator given in Section 2.1, where we denoted the flow by φ_h^G. Now, solving the low-rank differential equations by applying the Strang splitting scheme we obtain the approximate solution

$$\mathbf{Y}(t) \approx \left(\Phi_{h/2}^{\mathbf{B}} \circ \varphi_h^G \circ \Phi_{h/2}^{\mathbf{B}}\right)(\mathbf{Y}^0).$$

In order to give error bounds of the low-rank Strang splitting scheme, we follow the strategy of proving Theorem 3.2 by bounding the error of the full-rank Strang scheme and its difference to the low-rank Strang splitting method. Imposing the same assumptions for the second order case, the error estimate of the full-rank Strang splitting conforms to the first order splitting integrator, see Proposition 3.3. On the other hand, comparing the full-rank Strang splitting with its low-rank variant, we can simplify the error analysis by employing Assumption 3.1*(1)* and find the local bound

$$\begin{aligned}
&\left\| \left(\Phi_{h/2}^{\mathbf{B}} \circ \Phi_h^G \circ \Phi_{h/2}^{\mathbf{B}}\right)(\mathbf{Y}^{k-1}) - \left(\Phi_{h/2}^{\mathbf{B}} \circ \varphi_h^G \circ \Phi_{h/2}^{\mathbf{B}}\right)(\mathbf{Y}^{k-1}) \right\| \\
&= \left\| \left(\Phi_{h/2}^{\mathbf{B}} \circ \left(\Phi_h^G - \varphi_h^G\right) \circ \Phi_{h/2}^{\mathbf{B}}\right)(\mathbf{Y}^{k-1}) \right\| \\
&\leq \mathrm{e}^{(h/2)\omega} \|(\Phi_h^G - \varphi_h^G)(\mathbf{Y}^{k-1})\|.
\end{aligned}$$

We observe that this is of the same form as the estimate in (3.22) within the proof of Lemma 3.5 and hence we have to analyze the error of the projector-splitting integrator. Since we require the same assumptions, the analysis of this local bound is performed analogously as in Lemma 3.5, i.e., in fact we have to bound the error of the projector-splitting integrator. Now, the analysis of the Strang splitting integrator in [KLW16] as mentioned in Theorem 2.4 in Section 2.3, only shows that the Strang splitting is also of order one in the presence of small singular values. Therefore, collecting the orders of the full-rank Strang and the low-rank Strang splitting for our problem yields a total error of order one in time.

3.4 Differential Lyapunov equation

As a special case of the class of stiff matrix differential equations given in (3.1), we consider differential Lyapunov equations (DLEs). For DLEs, the term $G(t, \mathbf{A}(t))$ in (3.1) is solution independent. We denote the resulting time-dependent matrix as $\mathbf{Q}(t)$. This gives us the DLE

$$\dot{\mathbf{A}}(t) = \mathbf{B}\,\mathbf{A}(t) + \mathbf{A}(t)\,\mathbf{B}^\top + \mathbf{Q}(t), \quad \mathbf{A}(t_0) = \mathbf{A}^0, \tag{3.26}$$

where $\mathbf{B}, \mathbf{Q}(t) \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_1}$ solves (3.26). Its exact solution can be determined by the variation-of-constans formula and is given by

$$\mathbf{A}(t) = \mathrm{e}^{(t-t_0)\,\mathbf{B}}\,\mathbf{A}(t_0)\mathrm{e}^{(t-t_0)\,\mathbf{B}^\top} + \int_{t_0}^{t} \mathrm{e}^{(t-s)\,\mathbf{B}}\,\mathbf{Q}(s)\mathrm{e}^{(t-s)\,\mathbf{B}^\top}\,\mathrm{d}s.$$

In order to find a low-rank approximation $\mathbf{Y}(t) \in \mathcal{M}$ for the solution $\mathbf{A}(t)$ of the DLE, we follow the procedure described in Section 3.1. First, we split the DLE into the following two subproblems,

$$\begin{aligned} \dot{\mathbf{A}}_1(t) &= \mathbf{B}\,\mathbf{A}_1(t) + \mathbf{A}_1(t)\,\mathbf{B}^\top, \qquad & \mathbf{A}_1(t_0) &= \mathbf{A}_1^0, \\ \dot{\mathbf{A}}_2(t) &= \mathbf{Q}(t), & \mathbf{A}_2(t_0) &= \mathbf{A}_2^0, \end{aligned} \tag{3.27}$$

which, after applying the full-rank Lie–Trotter splitting integrator presented in Section 3.1.1, results in the approximate solution

$$\mathbf{A}(t) \approx (\Phi_h^{\mathbf{B}} \circ \Phi_h^{\mathbf{Q}})(\mathbf{A}^0),$$

where we have chosen $\mathbf{A}^0 = \mathbf{A}_2^0$ for the initial value and denoted the solution operator of the second subproblem (3.27) as $\Phi_h^{\mathbf{Q}}$. Then, as described in Section 3.1.2, we follow the dynamical low-rank approach and project the two arising subproblems orthogonally onto the tangent space, which yields

$$\begin{aligned} \dot{\mathbf{Y}}_1(t) &= \mathbf{B}\,\mathbf{Y}_1(t) + \mathbf{Y}_1(t)\,\mathbf{B}^\top, \qquad & \mathbf{Y}_1(t_0) &= \mathbf{Y}_1^0, \\ \dot{\mathbf{Y}}_2(t) &= \mathrm{P}(\mathbf{Y}_2(t))\,\mathbf{Q}(t), & \mathbf{Y}_2(t_0) &= \mathbf{Y}_2^0\,. \end{aligned} \tag{3.28}$$

We continue along the low-rank Lie–Trotter splitting integrator and solve the second subproblem by the projector-splitting integrator, where we denote the solution operator as $\varphi_h^{\mathbf{Q}}$. This results in a low-rank approximation

$$\mathbf{Y}(t) \approx (\Phi_h^{\mathbf{B}} \circ \varphi_h^{\mathbf{Q}})(\mathbf{Y}^0),$$

where $\mathbf{Y}^0$ is a rank r approximation to $\mathbf{A}^0$. The analysis of the global error of this scheme for the DLE goes along the proofs performed in Section 3.2, if the DLE satisfies Assumption 3.1. Hence, in order to give error bounds for the above proposed procedure for DLEs, we simply verify that the DLE fulfills each condition of this assumption.

(1) In the framework of DLEs, the matrix $\mathbf{B} \in \mathbb{R}^{n_1 \times n_1}$ comes from a spatial discretization of a superordinate parabolic partial differential equation and generates a strongly continuous semigroup. Now, by [EN99, Proposition 5.5] we conclude that the matrix $\mathbf{B}$ fulfills the exponential boundedness as well as the parabolic smoothing property in this assumption.

(2) Since the matrix $\mathbf{Q}(t)$ is independent of the exact solution $\mathbf{A}(t)$, the derivative with respect to the solution always exists and is continuous. Therefore $\mathbf{Q}(t)$ is continuously differentiable. This implies that $\mathbf{Q}(t)$ is bounded by B and Lipschitz continuous with Lipschitz constant $L = 0$.

(3) According to Assumption 3.1*(3)*, we assume that $\mathbf{Q}(t)$ is in the tangent space $\mathcal{T}_{\mathbf{Y}_2}\mathcal{M}$ up to a small perturbation. To make sure that $\mathbf{Q}(t)$ complies with this assumption, we rewrite

$$\mathbf{Q}(t) = \mathbf{Q}(t) - \mathrm{P}(\mathbf{Y}_2)\,\mathbf{Q}(t) + \mathrm{P}(\mathbf{Y}_2)\,\mathbf{Q}(t),$$

where we identify

$$\mathrm{P}(\mathbf{Y}_2)\,\mathbf{Q}(t) =: M(t,\mathbf{Y}_2) \in \mathcal{T}_{\mathbf{Y}_2}\mathcal{M} \quad \text{and} \quad \mathbf{Q}(t) - \mathrm{P}(\mathbf{Y}_2)\,\mathbf{Q}(t) =: R(t,\mathbf{Y}_2),$$

with

$$\begin{aligned}\|\mathbf{R}(t,\mathbf{Y}_2)\| &= \|(\mathbf{I} - \mathrm{P}(\mathbf{Y}_2))\,\mathbf{Q}(t)\| \\ &\leq \|\mathbf{I} - \mathrm{P}(\mathbf{Y}_2)\|\|\mathbf{Q}(t)\| \\ &\leq \sup_{t\in[t_0,T]} \|\mathbf{Q}(t)\| \\ &\leq \varepsilon,\end{aligned}$$

i.e., ε depends on the norm of $\mathbf{Q}(t)$. After this adaption, we can write $\mathbf{Q}(t) = M(t,\mathbf{Y}_2) + R(t,\mathbf{Y}_2)$, where $\mathbf{Q}(t)$ artificially depends on the approximate solution $\mathbf{Y}_2$.

(4) The condition about the initial value cannot be approved here, but we simply assume that the initial value $\mathbf{A}^0$ of the DLE and the rank r initial value $\mathbf{Y}^0$ are δ-close. In practice, this is assured by performing a truncated SVD of the initial matrix $\mathbf{A}^0$, which results in a best-approximation that we take for $\mathbf{Y}^0$, see the result of Eckart and Young in [EY36].

Hence, the DLE as a special case of a stiff differential equation of type (3.1) fulfills the conditions of Assumption 3.1 and therefore the general framework for an error analysis proposed in Section 3.2 is suitable for the analysis of the corresponding low-rank Lie–Trotter integrator for the DLE. Thus, we can simply adapt the error analysis given in Section 3.2.

With regard to the full-rank Lie–Trotter splitting error, we observe that splitting the DLE (3.26) into two subproblems is not affected by the modification $G(t,\mathbf{Y}_2(t)) = \mathbf{Q}(t)$. Hence, the structure of the global error bound for $E^n_{\mathrm{sp}} = \mathbf{A}(t_n) - (\Phi^{\mathbf{B}}_h \circ \Phi^G_h)(\mathbf{A}^0)$ is the same as in Proposition 3.3. What differs are the appearing constants, since the Lipschitz constant vanishes in our situation. This results in

$$\|E^n_{\mathrm{sp}}\| = \|\mathbf{A}(t_0 + nh) - (\Phi^{\mathbf{B}}_h \circ \Phi^G_h)^n(\mathbf{A}^0)\| \leq C(\omega,B,T)h(1 + |\log(h)|),$$

where the constant is independent of L.

Furthermore, for the distance between the full-rank Lie–Trotter splitting and its low-rank variant, the modification of the right-hand side of the second subproblem being independent of the solution has an impact on the given problem itself, though the error

analysis as in Proposition 3.4 keeps its procedure. Therefore, taking into account that in the situation of the DLE, the Lipschitz constant is $L = 0$, the bound E_{lr}^n is reduced to

$$\|E_{\mathrm{lr}}^n\| = \|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{Y}^0) - (\Phi_h^{\mathbf{B}} \circ \varphi_h^G)^n(\mathbf{Y}^0)\| \leq C_1 \varepsilon,$$

with

$$C_1 = 7\mathrm{e}^{(T-t_0)\omega} \frac{\mathrm{e}^{(T-t_0)\omega} - 1}{\omega}.$$

Note that this bound is independent of the time step size h, since the constant C_2 from Proposition 3.4 vanishes.

Finally, the error of the propagated initial distance $\|\mathbf{A}^0 - \mathbf{Y}^0\|$ does not depend on L, but keeps its form, i.e. we find

$$\|E_\delta^n\| = \|(\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{A}^0) - (\Phi_h^{\mathbf{B}} \circ \Phi_h^G)^n(\mathbf{Y}^0)\| \leq c_0 \delta,$$

where

$$c_0 = \mathrm{e}^{\omega(T-t_0)}.$$

3.5 Differential Riccati equation

Classical representatives of stiff matrix differential equations of type (3.1) are Differential Riccati Equations (DREs), which exhibit the same form of stiffness, but have a specific nonlinearity. They are of the form

$$\dot{\mathbf{A}}(t) = \mathbf{B}\,\mathbf{A}(t) + \mathbf{A}(t)\,\mathbf{B}^\top + \mathbf{Q}(t) - \mathbf{A}(t)\mathbf{N}\,\mathbf{A}(t), \quad \mathbf{A}(t_0) = \mathbf{A}^0, \tag{3.29}$$

where $\mathbf{B}, \mathbf{Q}(t), \mathbf{N}, \mathbf{A}(t) \in \mathbb{R}^{n_1 \times n_1}$ and the nonlinear term is quadratic. The exact solution of the DRE (3.29) is given by

$$\mathbf{A}(t) = \mathrm{e}^{(t-t_0)\,\mathbf{B}}\,\mathbf{A}(t_0)\mathrm{e}^{(t-t_0)\,\mathbf{B}^\top} + \int_{t_0}^{t} \mathrm{e}^{(t-s)\,\mathbf{B}}(\mathbf{Q}(s) - \mathbf{A}(s)\mathbf{N}\,\mathbf{A}(s))\mathrm{e}^{(t-s)\,\mathbf{B}^\top}\,\mathrm{d}s.$$

We follow the procedure of the integrator presented in Section 3.1, where we first split the DRE into a stiff linear part and a non-stiff nonlinear differential equation. Afterwards, we follow the approach of dynamical low-rank approximation, which results in the differential equations

$$\begin{aligned}
\dot{\mathbf{Y}}_1(t) &= \mathbf{B}\,\mathbf{Y}_1(t) + \mathbf{Y}_1(t)\,\mathbf{B}^\top, & \mathbf{Y}_1(t_0) &= \mathbf{Y}_1^0, \\
\dot{\mathbf{Y}}_2(t) &= \mathrm{P}(\mathbf{Y}_2(t))\,(\mathbf{Q}(t) - \mathbf{Y}_2(t)\mathbf{N}\,\mathbf{Y}_2(t))\,, & \mathbf{Y}_2(t_0) &= \mathbf{Y}_2^0\,.
\end{aligned}$$

We continue following the procedure of the presented integrator: we apply the projector-splitting integrator to obtain a low-rank approximation to $\mathbf{Y}_2(t)$ and solve the first sub-problem directly.

The error analysis of the global error of the low-rank Lie–Trotter splitting integrator in the situation of the DRE requires a review of Assumption 3.1, which will be done in the following:

(1) In the framework of the DRE, the matrix $\mathbf{B} \in \mathbb{R}^{n_1 \times n_1}$ comes from a spatial discretization of a superordinate parabolic partial differential equation and generates a strongly continuous semigroup, see, e.g., [EN99]. Therefore, $\mathbf{B}$ fulfills the required bounds in condition *(1)*.

(2) Since the matrix $\mathbf{Q}(t)$ is independent of the exact solution $\mathbf{A}(t)$, we require the nonlinear part $\mathbf{A}(t)\mathbf{N}\,\mathbf{A}(t)$ to be continuously differentiable, such that in total the full right-hand side $\mathbf{Q}(t) - \mathbf{A}(t)\mathbf{N}\,\mathbf{A}(t)$ is continuously differentiable in the neighborhood of the exact solution $\mathbf{A}(t)$. Therefore, the derivative with respect to the solution exists and is continuous. Hence, $\mathbf{Q}(t) - \mathbf{A}(t)\mathbf{N}\,\mathbf{A}(t)$ is bounded by B and Lipschitz continuous with Lipschitz constant $L > 0$.

(3) The nonlinear term within the DRE is supposed to be in the tangent space up to a small remainder. To adapt the quadratic term to this condition, we identify

$$M(t, \mathbf{Y}_2) = \mathrm{P}(\mathbf{Y}_2)\,\mathbf{Q}(t) - \mathbf{Y}_2\,\mathbf{N}\,\mathbf{Y}_2, \qquad \text{and} \qquad R(t, \mathbf{Y}_2) = \mathbf{Q}(t) - \mathrm{P}(\mathbf{Y}_2)\,\mathbf{Q}(t),$$

with $\|R(t, \mathbf{Y}_2)\| \leq \varepsilon$, such that we rewrite

$$\mathbf{Q}(t) - \mathbf{Y}_2(t)\mathbf{N}\,\mathbf{Y}_2(t) = M(t, \mathbf{Y}_2) + R(t, \mathbf{Y}_2).$$

Here, the remainder $R(t, \mathbf{Y}_2)$ is in the complement of $\mathcal{T}_{\mathbf{Y}_2}\mathcal{M}$. Recognizing that $M(t, \mathbf{Y}_2)$ as identified above is in the tangent space $\mathcal{T}_{\mathbf{Y}_2}\mathcal{M}$ requires more care. Recall that the term $M(t, \mathbf{Y}_2)$ is the sum of two elements of the tangent space. Now, $\mathrm{P}(\mathbf{Y})Q(t)$ is an element of the tangent space because of the orthogonal projection onto it. For $\mathbf{Y}_2(t)\mathbf{N}\,\mathbf{Y}_2(t)$ we make use of the explicit form of the splitted projection within the projector-splitting integrator. Since $\mathbf{Y}_2(t) \in \mathcal{M}$, we can factorize this matrix into $\mathbf{Y}_2 = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top$ and then observe that

$$\begin{aligned}
\mathrm{P}(\mathbf{Y}_2(t))(\mathbf{Y}_2\,\mathbf{N}\,\mathbf{Y}_2) &= \mathbf{U}\,\mathbf{U}^\top(\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top\,\mathbf{N}\,\mathbf{Y}_2) - \mathbf{U}\,\mathbf{U}^\top(\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top\,\mathbf{N}\,\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top)\,\mathbf{V}\,\mathbf{V}^\top \\
&\quad + (\mathbf{Y}_2\,\mathbf{N}\,\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top)\,\mathbf{V}\,\mathbf{V}^\top \\
&= \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top\,\mathbf{N}\,\mathbf{Y}_2 - \mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top\,\mathbf{N}\,\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top + \mathbf{Y}_2\,\mathbf{N}\,\mathbf{U}\,\mathbf{S}\,\mathbf{V}^\top \\
&= \mathbf{Y}_2\,\mathbf{N}\,\mathbf{Y}_2,
\end{aligned}$$

where we have used the fact that $\mathbf{U}$ and $\mathbf{V}$ have orthonormal columns. Since $\mathcal{T}_{\mathbf{Y}_2}\mathcal{M}$ is a vector space we conclude that $M(t, \mathbf{Y}_2) \in \mathcal{T}_{\mathbf{Y}_2}\mathcal{M}$.

(4) Instead of verifying the condition about the δ-distance of the initial value $\mathbf{A}^0$ and the starting value $\mathbf{Y}^0$, we assume that this condition holds true. This is a reasonable assumption due to the available SVD of $\mathbf{A}^0$, which results in a rank r best approximation $\mathbf{Y}^0$.

Since the DRE fulfills each condition of Assumption 3.1, we conclude that the global error of the low-rank Lie–Trotter splitting integrator when applied onto (3.29) is the same as stated in Theorem 3.2.

We have splitted the DRE into a linear and a nonlinear part. There exist other splitting methods that separate the DRE into a DLE plus the nonlinear term. A convergence analysis for such a splitting method in the setting of Hilbert–Schmidt operators was proposed in [HS14]. Though it is not proven, it is expected that the way of splitting as in [HS14] would also suit the dynamical low-rank approximation of the DRE, since one subproblem would simply correspond to the DLE, where the dynamical low-rank approximation is described in Section 3.4. The second subproblem would consist of the nonlinearity of the DRE and this will be dealt by the dynamical low-rank approach as described in Section 3.1 for the nonlinear term. However, we do not amplify this way of splitting, since it requires more computational effort due to the nested DLE within the first subproblem, which then in turn has to be solved by the proposed method and cannot be solved directly (as is the case for the linear part, see Section 3.1.1).

Moreover, a low-rank second-order splitting method is proposed in [Sti15] in the large-scale case and under the additional assumption that the matrices $\mathbf{N}$ and $\mathbf{Q}(t)$ are both, symmetric and positive definite for all $t_0 \leq t \leq T$.

3.6 Numerical examples

We now turn to numerical experiments that corroborate the favorable behavior of the low-rank Lie–Trotter splitting method proposed in Section 3.1 when computing low-rank approximations of semilinear stiff differential equations. The numerical examples we present here are taken from [OPW18].

We have seen in the error analysis of the Lie–Trotter splitting method that the error bounds do not depend on singular values. The first numerical example demonstrates the robustness of the method with respect to that. Also, it shows first order convergence of the method when applied to a stiff matrix differential equation without step size restriction due to the Lipschitz constant of the stiff right-hand side.

The aim of the second numerical experiment is to illustrate the performance of the low-rank Lie–Trotter splitting proposed in Section 3.1 and to show consistency with the convergence result of Theorem 3.2. There, the underlying differential equation is a DRE.

3.6.1 A reaction-diffusion equation

The main advantage of the integration method we propose is its insensitivity against stiffness. We illustrate this favorable behavior with the help of an example.

Consider the following two-dimensional partial differential equation

$$\partial_t u = \alpha \Delta u + u^3, \qquad u(0, x, y) = 16\, x(1-x)y(1-y),$$

where $\alpha = 1/50$. We solve this problem on the spatial domain Ω to be the unit square, subject to homogeneous Dirichlet boundary conditions, for times $0 \leq t \leq T$. Using second order finite differences, we discretize this partial differential equation in space with m inner points in each direction and denote the grid size by h, which is $h = \frac{1}{m+1}$. The inner grid

points in x and y direction are denoted by

$$x_i = ih \quad \text{and} \quad y_j = jh \quad \text{for} \quad 1 \leq i, j \leq m,$$

respectively. Denoting the one-dimensional stencil matrix in x direction by $\mathbf{B}$, this results in the matrix differential equation

$$\dot{\mathbf{A}}(t) = \alpha \mathcal{B}[\mathbf{A}(t)] + \mathbf{A}(t)^3, \qquad \mathbf{A}(t_0) = \mathbf{A}^0,$$

where $\mathcal{B}[\mathbf{A}(t)] = \mathbf{B}\,\mathbf{A}(t) + \mathbf{A}(t)\,\mathbf{B}^\top$ and $\mathbf{A}(t) \in \mathbb{R}^{m \times m}$. This stiff differential equation is of the form we have studied in Section 3.1, where the stiff spatial discretization $\mathcal{B}[\mathbf{A}(t)]$ of the elliptic operator within the PDE is induced by the matrix $\mathbf{B}$. The matrix $\mathbf{A}_{ij}(t)$ is the sought after approximation of $u(t, x_i, y_j)$, $1 \leq i, j \leq m$. The nonlinearity is realized by an entrywise product.

In our numerical experiment we choose $m = 500$. The reference solution is computed with DOPRI5, a Runge–Kutta method of order 5 with adaptive step size strategy and with high precision, see [HNW93, II.5]. The first 30 singular values of the reference solution at final time $T = 0.5$ are decaying exponentially. Figure 3.2 shows that the effective rank is about $r = 10$ and the remaining singular values are negligible small:

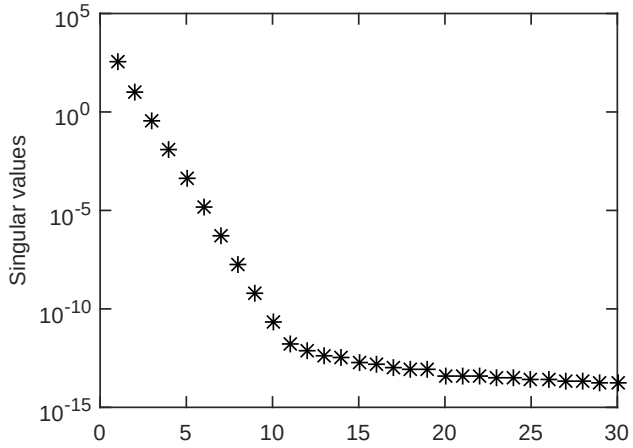

Figure 3.2: Singular values of the reference solution, which is computed with DOPRI5.

In the situation of small singular values, we compute the low-rank approximation matrix $\mathbf{Y}$ for different time steps and different approximation ranks. The error of the low-rank Lie–Trotter splitting integrator compared to the reference solution is given in Figure 3.3, where we measure the error in the Frobenius norm.

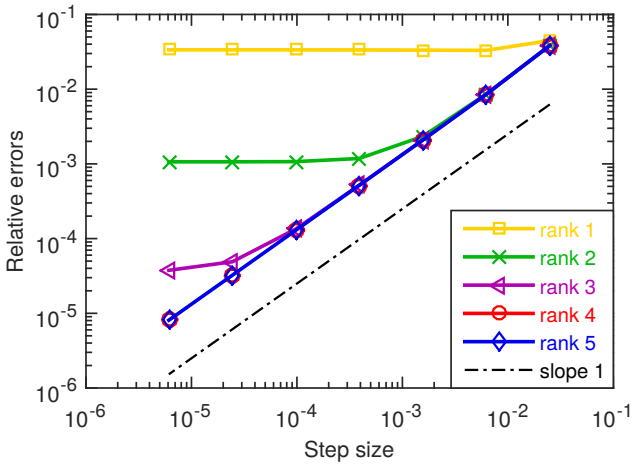

Figure 3.3: Error of our proposed first-order splitting method as a function of the step size and the approximation rank.

We observe an explicit dependence of the error on the rank and on the step size. If the approximation rank is chosen sufficiently large (rank 4 and 5) we solely observe the first-order error due to the splitting into the linear and the nonlinear subproblem. On the other hand, a bad choice of the approximation rank (rank 1, 2 and 3) leads to a stagnation of the error, independently on the refinement of the time step size, such that we can not verify first order convergence of the proposed method. Here, the difference E_{lr}^n between the full-rank Lie–Trotter and the low-rank Lie–Trotter is dominant. If this difference E_{lr}^n becomes small enough, we observe that the full-rank Lie–Trotter splitting, i.e., the outer Lie–Trotter splitting, is convergent of order one in time.

Further, the Lipschitz constant L of the right-hand side of the matrix differential equation for $\mathbf{A}(t)$ is proportional to αw^{-2}, where w is the mesh width of the spatial discretization. For our choice of parameters we have $L \approx 5 \cdot 10^{+3}$. We observe that our integrator gives good results not only for small, but also for large time step sizes.

3.6.2 A differential Riccati equation

We study a DRE arising in optimal control for linear quadratic regulator problems. In order to take an interesting application into account, we use the numerical example presented in [HS14]. Thus we consider the linear control system

$$\dot{x} = \mathbf{B}\, x + v, \quad x(0) = x_0,$$

where $\mathbf{B} \in \mathbb{R}^{m \times m}$ is the system matrix, $x \in \mathbb{R}^m$ the state variable and $v \in \mathbb{R}^m$ the control. The functional $\mathcal{J}$ that has to be minimized is given by

$$\mathcal{J}(v, x) = \frac{1}{2} \int_0^T \Big(x(t)^\top \mathbf{C}^\top \mathbf{C}\, x(t) + v(t)^\top v(t) \Big)\, \mathrm{d}t,$$

where $\mathbf{C} \in \mathbb{R}^{q\times m}$. Further, the optimal control is given in feedback form by $v_{\text{opt}}(t) = -\mathbf{A}(t)x(t)$, where $\mathbf{A}(t)$ is the solution of the following DRE

$$\dot{\mathbf{A}}(t) = \mathbf{B}^\top \mathbf{A}(t) + \mathbf{A}(t)\,\mathbf{B} + \mathbf{C}^\top \mathbf{C} - \mathbf{A}(t)^2, \tag{3.30}$$

which is of the form (3.29) with $\mathbf{Q} = \mathbf{C}^\top \mathbf{C}$ and $\mathbf{N} = \mathbf{I}_{n_1}$ being the identity matrix.

The matrix $\mathbf{B}$ arises from the spatial discretization of the diffusion operator

$$\mathcal{D} = \partial_x\left(\alpha(x)\partial_x(\cdot)\right) - \lambda\,\mathbf{I},$$

defined on the spatial domain $\Omega = (0,1)$ subject to homogeneous Dirichlet boundary conditions. We choose $\alpha(x) = 2 + \cos(2\pi x)$ and $\lambda = 1$. The finite difference discretization of the operator $\mathcal{D}$ satisfies Assumption 3.1*(1)*. Now, let q be an odd, positive number. The matrix $\mathbf{C} \in \mathbb{R}^{q\times m}$ is defined by q independent vectors $\{1, g_1, \ldots, g_{(q-1)/2}, f_1, \ldots, f_{(q-1)/2}\}$, where

$$g_k(x) = \sqrt{2}\cos(2\pi k x) \quad \text{and} \quad f_k(x) = \sqrt{2}\sin(2\pi k x), \quad k = 1, \ldots, (q-1)/2.$$

We implement this example by choosing $\mathbf{A}^0 = \mathbf{0}$ as initial value, $T = 0.1$ as final computation time, $m = 200$ mesh points and $q = 9$ independent vectors to build up the matrix $\mathbf{C}$. We apply the DOPRI5 method, which is a Runge–Kutta method of order 5, in order to determine a reference solution for the DRE (3.30).

Studying the effective rank of the reference solution in Figure 3.4, we observe that it changes at the beginning in time, but settles down to around $r = 30$ shortly after starting the computation. This is visualized in the left picture of Figure 3.4.

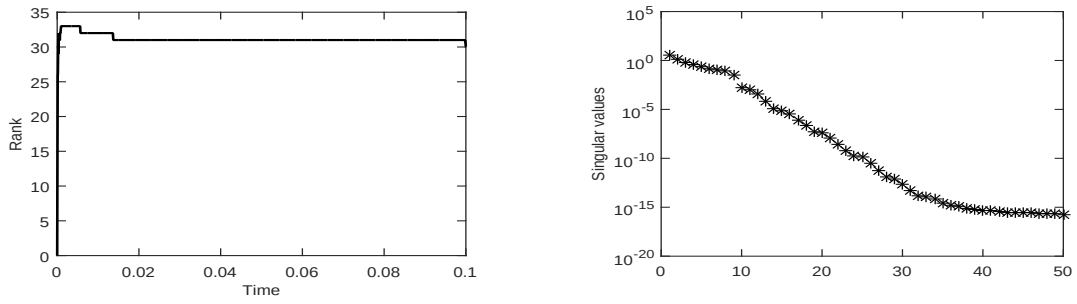

Figure 3.4: Singular values of the reference solution. Left: Rank of the reference solution as a function of time. Right: First 50 singular values of the reference solution at $T = 0.1$.

Within the same figure, but on the right-hand side, we see that the singular values decay and starting from the 31st singular value, the remaining ones are of size $\approx 10^{-15}$ and are therefore negligible from the numerical perspective. Hence, the first 30 singular values have to be taken into account. This is consistent with the figure on the left.

We further observe on the right-hand side in Figure 3.4 a gap between the ninth and the tenth singular value of the reference solution at final time $T = 0.1$.

The effective rank of the solution stays low during the evolution in time. Nevertheless, the low-rank Lie–Trotter splitting integrator is not affected by the small singular values.

This favorable behavior is corroborated in Figure 3.5, which shows the error behavior of the low-rank Lie–Trotter splitting proposed in Section 3.1.

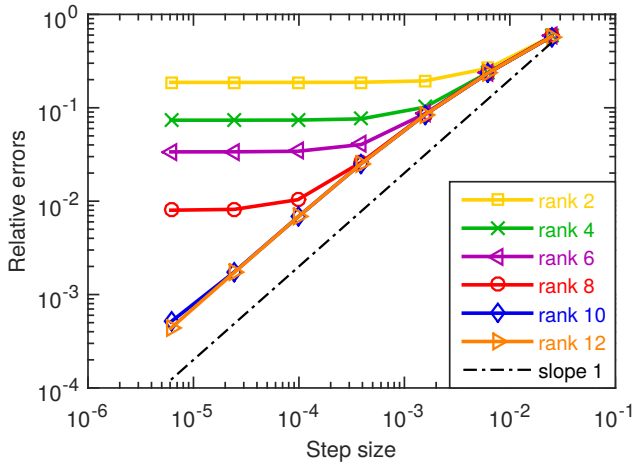

Figure 3.5: Errors of the low-rank Lie–Trotter splitting in the Frobenius norm as a function of step size and rank at $T = 0.1$ for the considered DRE for $m = 200$.

We further observe that the error is composed by two different contributions. The choice of a (too) small approximation rank (rank $2, 4, 6, 8$) results in stagnation of the error, since then the difference between the full-rank splitting solution and the low-rank splitting solution becomes larger than the error of the full-rank Lie–Trotter splitting and therefore we do not observe convergence of order one. On the other hand, if this difference E_{lr}^n becomes small enough, we observe that the full-rank Lie–Trotter splitting, i.e., the outer Lie–Trotter splitting, is convergent of order one in time (rank $10, 12$). This is consistent with the convergence result given in Theorem 3.2.

Studying Figure 3.5, we would expect the curve for rank 10 to adjust to the trend of behavior of the curves for the other depicted ranks, i.e., that it would lead to an error of about 10^{-3} for the time step size 10^{-5} and then starting from step size about $0.5 \cdot 10^{-5}$ adapt to the error performance of the method for rank 12. We do not observe this behavior here. This can be explained with regard to the gap between the ninth and the tenth singular value mentioned above and observed on the right-hand side in Figure 3.4. Due to this gap, the error behavior of the low-rank Lie–Trotter splitting integrator passes directly to the convergence of order one.

4 Time integration of rank-constrained Tucker tensors

In this chapter, we increase the order of the object that needs to be approximated and extend the matrix case to tensors $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ with $t_0 \leq t \leq T$ and $d \geq 3$. This tensor is either given explicitly, or it is the unknown solution to a *tensor differential equation*

$$\dot{A}(t) = F(t, A(t)), \qquad A(t_0) = A^0, \tag{4.1}$$

where $\dot{A}(t) = \mathrm{d}A/\mathrm{d}t$. As in the previous chapters, we search a low-rank approximation to the solution of (4.1), but with the difference that the search space now is given by the manifold

$$\mathcal{M} = \{Y(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d} \mid \operatorname{rank} Y(t) = (r_1, \ldots, r_d)\}$$

of tensors of low multilinear rank $r := (r_1, \ldots, r_d) \in \mathbb{N}^d$. Following the basic concept of dynamical low-rank approximation presented in Chapter 1, we evolve a tensor ODE for the approximation tensor, which is given by

$$\dot{Y}(t) = \mathrm{P}(Y)F(t, Y(t)), \qquad Y(t_0) = Y^0. \tag{4.2}$$

For solving this differential equation, we choose the Tucker tensor format as a low-rank representation of the approximation tensor. We extend the matrix projector-splitting integrator to Tucker tensors by proposing an efficient integrator, based on the idea of an inexact solution of substeps within the matrix projector-splitting integrator applied to matricizations of the current (intermediate) approximation tensor in Section 4.2. This derivation allows us to transfer the error analysis as well as the exactness property of the matrix integrator to the tensor case, see Section 4.5 and Section 4.4, respectively. Afterwards, in Section 4.6, we will deal with another integration method for Tucker tensors, which was presented in [Lub15] and recalled in Section 4.6.1. We will give a new proof of exactness of this integrator in Section 4.6.3, which is neither published nor submitted elsewhere. Sections 4.6.4 and 4.6.5 are concerned with the relation between the nested Tucker integrator and the alternative integration method, where we first discuss what they have in common and in which sense they differ, and second we prove their mathematical equivalence. At the end of this chapter, we provide two numerical examples, which illustrate the error behavior of the newly proposed nested integration method for Tucker tensors.

The main source we draw the material of Sections 4.2 - 4.5 as well as of Section 4.7 from is [LVW18], which is a published article of the author in collaboration with Ch. Lubich and B. Vandereycken. Section 4.6 is partly taken from [Lub15] and partly new material, which is pointed out as such. We begin with introducing the Tucker tensor format and explain some calculus that is needed for our purposes, as well as how to compute the Tucker representation for a given tensor in Section 4.1. Here, we lean on the fundamental survey article [KB09] of T.G. Kolda and B.W. Bader about tensor decompositions, on the original publication [Tuc66] by L.R. Tucker, where he proposed the tensor format which nowadays is named after him as well as on the work [DLDMV00a] by L. De Lathauwer, B. De Moor and J. Vandewalle that introduces the multilinear singular value decomposition and deals with computational aspects.

4.1 Tucker tensor format

The Tucker tensor decomposition is the format of choice, which we will use as a low-rank representation of a given tensor $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ in the subsequent sections. It factorizes A into a product of a tensor multiplied by matrices along each mode i, where $i = 1, \ldots, d$.

The main part of this section is recalled from [KB09, Tuc66, DLDMV00a] and occasionally we will follow different publications, which we then name as such.

4.1.1 Modal multiplication of a tensor by a matrix

In order to present the Tucker tensors, we need to introduce a tensor-matrix product. In case when the tensor is of dimension two, we are back in the matrix-matrix product explained in Section 1.3.3, which is the inner product of the rows of the first matrix with the columns of the second. Now, suppose the tensor $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is of order $d \geq 3$, then there are several choices of modes of the tensor that can be multiplied by the rows of the matrix. Hence, we need to specify the mode of the tensor, in which the multiplication by the matrix is performed. Here, we follow the survey article [KB09] about tensor decompositions. Given a tensor $G \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ and a matrix $\mathbf{W} \in \mathbb{R}^{m_i \times n_i}$, the *i-mode product* $G \times_i \mathbf{W}$ is defined elementwise as

$$(G \times_i \mathbf{W})_{j_1 \cdots j_{i-1} \mu_i j_{i+1} \cdots j_d} = \sum_{j_i=1}^{n_i} g_{j_1 \cdots j_d} w_{\mu_i j_i},$$

for all $\mu_i = 1, \ldots, m_i$ and $j_i = 1, \ldots, n_i$. Since the factor matrix operates on the i-th mode of the tensor, the dimension of the corresponding mode of the resulting tensor-matrix product has changed: the size then is $n_1 \times \cdots \times n_{i-1} \times m_i \times n_{i+1} \cdots \times n_d$.

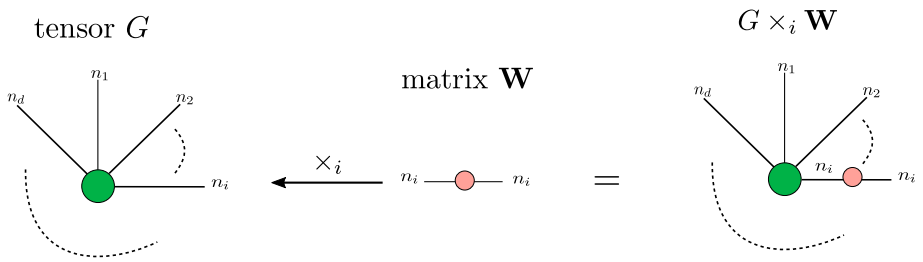

Figure 4.1: Illustration of the i-mode product of a tensor by a matrix. Here, the number of branches depict the number of free indices, or rather the dimension of the tensor. Since a matrix is a two-dimensional object, it has two branches. Now, we can think of multiplying a tensor by a matrix in the above sense by sticking the matrix on the tensor in the corresponding mode.

The i-mode product is invariant under the order of multiplication with respect to the factor matrices when multiplying in different modes $i \neq l$, i.e.,

$$G \times_i \mathbf{W} \times_l \overline{\mathbf{W}} = (G \times_i \mathbf{W}) \times_l \overline{\mathbf{W}} = (G \times_l \overline{\mathbf{W}}) \times_i \mathbf{W},$$

for $\mathbf{W} \in \mathbb{R}^{m_i \times n_i}, \overline{\mathbf{W}} \in \mathbb{R}^{m_l \times n_l}$. Analogously, since the factor matrices operate on the i-th and l-th mode, respectively, the resulting tensor is of size $n_1 \times \cdots \times n_{i-1} \times m_i \times n_{i+1} \times \cdots \times n_{l-1} \times m_l \times n_{l+1} \times \cdots \times n_d$. In case we multiply the matrices $\mathbf{W}$ and $\overline{\mathbf{W}}$ by a tensor in the same mode, the order of the multiplication does matter, and computing the product is only possible if $m_i = n_l$, i.e.,

$$G \times_i \mathbf{W} \times_i \overline{\mathbf{W}} = G \times_i (\overline{\mathbf{W}} \cdot \mathbf{W}) \in \mathbb{R}^{n_1 \times \cdots \times n_{i-1} \times m_l \times n_{i+1} \times \cdots \times n_d}.$$

The procedure of the i-mode product is easier to understand in terms of its matricized version. In order to present the matrix form of the i-mode product, we explain how to convert a tensor into a matrix beforehand.

4.1.2 Transforming a tensor into a matrix

In practice, the i-mode product is not computed elementwise, but makes use of the *unfolding* of a tensor into a matrix, also known as *matricization*. There are several ways of how to transform a tensor into a matrix, which all follow the principle of partitioning the modes and reordering the entries of the tensor. Let $\mathcal{I} = \{1, \ldots, d\}$ be the set of modes and let the two disjoint subsets $\mathcal{R} = \{\rho_1, \ldots, \rho_i\} \subset \mathcal{I}$ and $\mathcal{C} = \{\gamma_1, \ldots, \gamma_{d-i}\} \subset \mathcal{I}$ be a partition of $\mathcal{I}$, such that $\mathcal{R} \cup \mathcal{C} = \mathcal{I}$. The set $\mathcal{R}$ determines the rows and the set $\mathcal{C}$ specifies the columns of the resulting matrix. Unfolding a tensor A of size $n_1 \times \cdots \times n_d$ yields a matrix of size $n_{\rho_1} \cdots n_{\rho_i} \times n_{\gamma_1} \cdots n_{\gamma_{d-i}}$, where its rows are generated by the indices in $\mathcal{R}$ and its columns are built from the indices in $\mathcal{C}$. We make this general concept more comprehensible on two special cases.

First, let us consider the two subsets of modes to be given as $\mathcal{R} = \{1, \ldots, i\}$ and $\mathcal{C} = \{i+1, \ldots, d\}$. Then, unfolding the tensor $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ gives the matrix

$$\mathbf{Unf}_i(A) \in \mathbb{R}^{n_1 \cdots n_i \times n_{i+1} \cdots n_d},$$

which we call the *i-th unfolding* of A. It aligns all entries $A(j_1, \dots, j_d)$ with fixed $j_1, \dots, j_i$ in a row of $\mathbf{Unf}_i(A)$ and orders rows and columns of $\mathbf{Unf}_i(A)$ in a colexicographical way.

We think about the i-th unfolding of a tensor as merging the first i indices, which then determines the rows and concatenating the remaining indices $i+1, \dots, d$ to form the columns of the resulting matrix, see Figure 4.2.

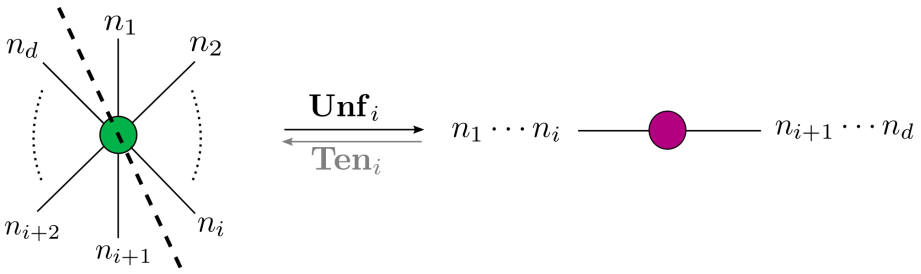

Figure 4.2: Illustration of the i-th unfolding of a tensor into a matrix. The dashed line depicts the partition of the indices of the tensor.

To clarify this procedure, we consider the following example. Let $A \in \mathbb{R}^{2\times4\times2\times2}$. Then, the 2-nd unfolding of A gives the matrix

$$\begin{array}{c} \\ (11) \\ (21) \\ (12) \\ (22) \\ (13) \\ (23) \\ (14) \\ (24) \end{array} \begin{array}{c} \begin{array}{cccc} (11) & (21) & (12) & (22) \end{array} \\ \begin{bmatrix} a_{1111} & a_{1121} & a_{1112} & a_{1122} \\ a_{2111} & a_{2121} & a_{2112} & a_{2122} \\ a_{1211} & a_{1221} & a_{1212} & a_{1222} \\ a_{2211} & a_{2221} & a_{2212} & a_{2222} \\ a_{1311} & a_{1321} & a_{1312} & a_{1322} \\ a_{2311} & a_{2321} & a_{2312} & a_{2322} \\ a_{1411} & a_{1421} & a_{1412} & a_{1422} \\ a_{2411} & a_{2421} & a_{2412} & a_{2422} \end{bmatrix} \end{array} = \mathbf{Unf}_2(A) \in \mathbb{R}^{8\times4}.$$

This type of unfolding is used in the numerical linear algebra literature for transforming tensor trains into matrices, see [Ose11], amongst others.

Second, let us consider the special case when $\mathcal{R} = \{\rho_1\}$ is a singleton and $\mathcal{C} = \{\gamma_1, \dots, \gamma_{d-1}\}$. This partition results in another important unfolding, which we will focus on throughout this chapter. Here, unfolding the tensor A results in the matrix

$$\mathbf{Mat}_i(A) \in \mathbb{R}^{n_i \times n_1 \cdots n_{i-1} n_{i+1} \cdots n_d},$$

which we denominate as *i-mode matricization*.

In terms of pictures, we think of the i-mode matricization as fixing the i-th mode of the tensor and folding the remaining indices together, see the figure below.

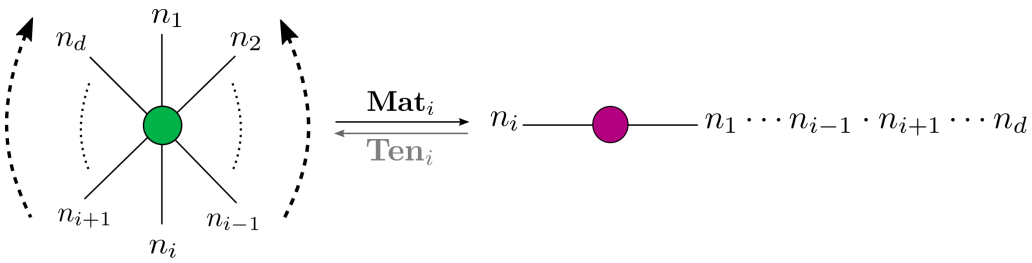

Figure 4.3: Illustration of the i-mode matricization of a tensor into a matrix. Here, the dashed arrows depict the folding of all indices $\neq i$ while the i-th index is fixed.

It arranges the entries of A that have the index $k = 1, \ldots, n_i$ in the i-th mode in the k-th row of the matrix $\mathbf{Mat}_i(A)$ in a colexicographical order. In the example above, this means that the 2-mode matricization of $A \in \mathbb{R}^{2\times4\times2\times2}$ results in the matrix

$$
\begin{array}{c} \\ (1) \\ (2) \\ (3) \\ (4) \end{array}
\begin{array}{c}
\begin{array}{cccccccc} (111) & (211) & (121) & (221) & (112) & (212) & (122) & (222) \end{array} \\
\begin{bmatrix}
a_{1111} & a_{2111} & a_{1121} & a_{2121} & a_{1112} & a_{2112} & a_{1122} & a_{2122} \\
a_{1211} & a_{2211} & a_{1221} & a_{2221} & a_{1212} & a_{2212} & a_{1222} & a_{2222} \\
a_{1311} & a_{2311} & a_{1321} & a_{2321} & a_{1312} & a_{2312} & a_{1322} & a_{2322} \\
a_{1411} & a_{2411} & a_{1421} & a_{2421} & a_{1412} & a_{2412} & a_{1422} & a_{2422}
\end{bmatrix}
\end{array}
= \mathbf{Mat}_2(A) \in \mathbb{R}^{4\times8}.
$$

We mention that it is also possible to unfold a tensor into a vector. This is the case when $\mathcal{C} = \emptyset$ and $\mathcal{R} = \mathcal{I}$. Vectorizing the tensor A gives $\mathbf{vec}(A) \in \mathbb{R}^{n_1 \cdots n_d \times 1}$, where again, the entries of the vector are ordered colexicographically. An illustration of the vectorization is given in the following figure:

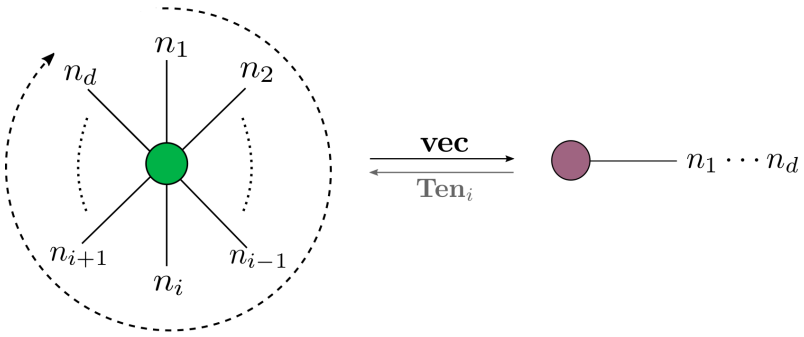

Figure 4.4: Illustration of vectorizing a tensor.

In the example above, vectorizing the tensor $A \in \mathbb{R}^{2\times4\times2\times2}$ results in

$$\begin{bmatrix} a_{1111} \\ a_{2111} \\ a_{1211} \\ \vdots \\ a_{2422} \end{bmatrix} = \mathbf{vec}(A) \in \mathbb{R}^{24\times1}.$$

Conversely, the tensor A can be reconstructed from its i-th unfolding, from its i-mode matricization and from its vectorization by simply rearranging the entries of the matrix/vector in the reverse way it was matricized/vectorized in the resulting tensor. Though we have introduced different denominations for the transformation into a matrix or a vector, we will use the same notation for tensorization, as it is automatically specified by the unfolding, matricization or vectorization. We write

$$A = \mathrm{Ten}_i(\mathbf{Unf}_i(A)), \qquad A = \mathrm{Ten}_i(\mathbf{Mat}_i(A)) \qquad \text{and} \qquad A = \mathrm{Ten}_i(\mathbf{vec}(A)),$$

respectively. In Figures 4.2, 4.3 and 4.4 the tensorization is depicted by the gray arrow.

In the literature, the ordering of the columns of the matricized tensor is inconsistent. However, this is not a difficulty within computations with matricized tensors, since once the choice of how to organize the columns of the resulting matrix is made, it simply has to be consistent during every computational step.

Since matricizing a tensor A into a matrix means reorganizing its elements, we can determine the norm of the tensor A by computing the norm of its matricization $\mathbf{Mat}_i(A)$. Now, decomposing $\mathbf{Mat}_i(A) = \mathbf{W}_i\,\mathbf{\Sigma}_i\,\widetilde{\mathbf{W}}_i^\top$ with orthonormal matrices $\mathbf{W}_i, \widetilde{\mathbf{W}}_i$, the Frobenius norm of A is determined by

$$\|A\|^2 = \|\mathbf{Mat}_i(A)\|^2 = \|\mathbf{\Sigma}_i\|^2, \qquad \text{for all } i = 1, \dots, d. \tag{4.3}$$

4.1.3 Tucker decomposition and its computation

With the i-mode tensor-matrix product and the concept of i-mode matricization at hand, we are now in the situation to present the Tucker decomposition and its matrix version.

Given a tensor $A \in \mathbb{R}^{n_1\times\cdots\times n_d}$, the *Tucker decomposition* factorizes A into a *core tensor* $G \in \mathbb{R}^{n_1\times\cdots\times n_d}$ and *factor matrices* $\mathbf{W}_i \in \mathbb{R}^{n_i\times n_i}$, with $i = 1, \dots, d$, such that each entry is determined by

$$a_{k_1,\dots,k_d} = \sum_{j_1=1}^{n_1} \cdots \sum_{j_d=1}^{n_d} g_{j_1,\dots,j_d} w_{k_1,j_1} \cdots w_{k_d,j_d}.$$

With $\times_i$ denoting a multilinear product of the core tensor by the matrices $\mathbf{W}_i$ along the i-th mode as introduced in Section 4.1.1, the shortform of the Tucker decomposition is given by

$$A = G \times_1 \mathbf{W}_1 \times_2 \cdots \times_d \mathbf{W}_d = G \bigtimes_{i=1}^{d} \mathbf{W}_i, \tag{4.4}$$

where $\bigtimes_{i=1}^{d}$ comprises the multilinear tensor-matrix product. Often, it is useful to work with matrices instead of tensors. To this end, we transform a Tucker tensor into a matrix. A matrix representation of size $\mathbb{R}^{n_i \times n_1 \cdots n_{i-1} n_{i+1} \cdots n_d}$ can be obtained by matricizing the core tensor in the i-th mode and multiplying it with the factor matrices by using the Kronecker product, such that

$$\mathbf{Mat}_i(A) = \mathbf{W}_i \ \mathbf{Mat}_i(G) \ \left(\mathbf{W}_1^\top \otimes \cdots \otimes \mathbf{W}_{i-1}^\top \otimes \mathbf{W}_{i+1}^\top \otimes \cdots \otimes \mathbf{W}_d^\top\right) \tag{4.5}$$

$$= \mathbf{W}_i \ \mathbf{Mat}_i(G) \ \bigotimes_{\substack{k=1 \\ k \neq i}}^{d} \mathbf{W}_k^\top, \tag{4.6}$$

where $\bigotimes_{k=1}^{d}$ denotes the Kronecker products in each mode. Note that Tucker [Tuc66] did not require any specific properties for the core tensor G and neither for the factor matrices $\mathbf{W}_i$, such as diagonality or orthonormality within the decomposition he presented. However, he gave an instruction of how to compute an exact Tucker decomposition (4.4) in the case when the factor matrices have orthonormal columns, see [Tuc66, Method 1], but did not specify how to compute the factor matrices exactly. De Lathauwer, De Moor and Vandewalle enhanced Tucker's method by making use of the SVD as a tool to decompose a matrix into a product of orthonormal factors, more precisely see Section 1.1. They call the method *higher order singular value decomposition* (HOSVD) [DLDMV00a], whose computation is given in Algorithm 4.

Algorithm 4: Tucker decomposition, Method 1/HOSVD

Data: Tensor $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$

Result: Tucker tensor $G \times_1 \mathbf{W}_1 \times_2 \cdots \times_d \mathbf{W}_d \in \mathbb{R}^{n_1 \times \cdots \times n_d}$

1 **begin**

2 **for** $i = 1$ **to** d **do**

3 compute singular value decomposition $\mathbf{Mat}_i(A) = \mathbf{W}_i \, \mathbf{\Sigma}_i \, \widetilde{\mathbf{W}}_i^\top$

4 set $G = A \times_1 \mathbf{W}_1^\top \times_2 \cdots \times_d \mathbf{W}_d^\top$

5 return G

6 return $\mathbf{W}_1, \ldots, \mathbf{W}_d$

The HOSVD of a d-th order tensor requires the SVD for determining the orthonormal factor matrices $\mathbf{W}_i$ for each mode $i = 1, \ldots, d$, and so it boils down to d matrix SVDs. For matrices, we know from Section 1.1 that a SVD exists. Therefore, we can always compute a SVD of each matricization of A and therefore, Algorithm 4 is a constructive proof of the existence of the Tucker decomposition. This result was proven in an analytical way in [DLDMV00a, Theorem 2], where it was shown that every tensor $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ can be represented in the Tucker format (4.4), where the factor matrices $\mathbf{W}_i$ are orthogonal $n_i \times n_i$ matrices and the core tensor G, which is of size $n_1 \times \cdots \times n_d$, can be determined as in line 4 in Algorithm 4. From now on, we make the convention that the factor matrices $\mathbf{W}_i$ are orthonormal for all $i = 1, \ldots, d$, although this is not required by the Tucker decomposition (4.4).

Not only the description of the algorithm, but also comparing the restated [DLDMV00a, Theorem 2] for the HOSVD with the corresponding result regarding the SVD, shows a clear analogy between the higher-dimensional and the two-dimensional case. The crucial difference is the *all-orthogonality* of the core tensor G, which means that fixing the i-th index and going through the resulting $(d-1)$-th order subtensors G_{i_a} for all $a = 1, \ldots, n_i$, it holds that

$$\langle G_{i_a}, G_{i_b} \rangle = 0, \qquad \text{for } a \neq b, \text{ and for all } i = 1, \ldots, d.$$

In case when A is a second order tensor, we can write the SVD of that matrix in terms of the i-mode product as

$$\mathbf{A} = \mathbf{W}\,\mathbf{\Sigma}\,\widetilde{\mathbf{W}}^{\top} = \mathbf{\Sigma} \times_1 \mathbf{W} \times_2 \widetilde{\mathbf{W}},$$

where $\mathbf{\Sigma} = G$, $\mathbf{W} = \mathbf{W}_1$ and $\widetilde{\mathbf{W}} = \mathbf{W}_2$ regarding the notation above. Here, the matrix $\mathbf{\Sigma}$ is diagonal, which is not required for $d \geq 3$. However, the matrix case is included in the tensor case regarding the property of being all-orthogonal, since the scalar product of two order-1 subtensors of the diagonal matrix $\mathbf{\Sigma}$ is zero.

Further, with the i-mode matricized decomposition (4.5) of the tensor A at hand, we can specify the norm of A given in (4.3). Since each factor matrix $\mathbf{W}_i$ is orthonormal, the *Frobenius norm* of $A = G \times_1 \mathbf{W}_1 \times_2 \cdots \times_d \mathbf{W}_d \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is

$$\|A\|^2 = \|\mathbf{Mat}_i(A)\|^2 = \Big\| \mathbf{W}_i \cdot \mathbf{Mat}_i(G) \cdot \bigotimes_{k \neq i} \mathbf{W}_k^{\top} \Big\|^2 = \|\mathbf{Mat}_i(G)\|^2 = \sum_{j_i=1}^{n_i} \big(\sigma_{j_i}^{(i)}\big)^2, \quad (4.7)$$

where $\sigma_{j_i}^{(i)}$ are the singular values of the matricized core $\mathbf{Mat}_i(G)$. Hence, the Frobenius norm for a tensor in the Tucker format is related to the Frobenius norm of a matrix with respect to the singular values.

Addressing ourselves back to the task of approximating an explicitly or implicitly given tensor of size $n_1 \times \cdots \times n_d$ in order to make computations with higher order tensors feasible, we observe that the Tucker decomposition as presented in (4.4) is not beneficial in this regard. To the contrary, assuming that $N = \max\{n_1, \ldots, n_d\}$, storing the given tensor would require saving about N^d entries and storing this tensor in the Tucker format requires the storage of $N^d + dN^2$ entries. This is due to the fact that the core tensor is of the same size as the given tensor and the factor matrices are of size $n_i \times n_i$. Another difficulty is the rank of the tensor in (4.4). We are seeking a low-rank approximation to the given tensor, which is supposed to be of multilinear low rank $(r_1, \ldots, r_d)$, but the rank of the Tucker decomposition (4.4) is not guaranteed to be low.

In view of line 3 in Algorithm 4, where the matrix SVD is involved, we overcome these difficulties by pursuing the idea of low-rank approximation for matrices, where the non-zero entries of the diagonal matrices $\mathbf{\Sigma}_i$ are limited to the desired rank r_i size by leaving the first r_i entries and setting the remaining ones to zero, such that $\mathbf{\Sigma}_i$ keeps its size. So in order to obtain a low multilinear rank $(r_1, \ldots, r_d)$ approximation to A, we restrict the non-zero entries of the core tensor G to the first r_i elements, for all $i = 1, \ldots, d$ and fill it up with zeros, such that the restricted core $\widehat{G}$ remains of size $n_1 \times \cdots \times n_d$. The orthonormal factor

matrices $\mathbf{W}_i$ are not modified. This method is known as *compact* or *truncated HOSVD* (THOSVD) and it results in an approximation tensor

$$\widehat{A} = \widehat{G} \times_1 \mathbf{W}_1 \times_2 \cdots \times_d \mathbf{W}_d \;\approx\; A. \tag{4.8}$$

We recall that discarding the smallest singular values within the matrix case yields a best rank r approximation to the considered matrix in terms of the low rank constraint, see the theorem of Eckart and Young mentioned in Section 1.1.

Unlike for the second order case, discarding the last $(n_i - r_i)$ singular values of the core for each mode $i = 1, \ldots, d$, does not result in the best approximation to A, see [DLDMV00a, Property 10].

However, the THOSVD (4.8) may lead to a good rank $(r_1, \ldots, r_d)$ approximation, since for a fixed mode i we have

$$\begin{aligned}
\|A - \widehat{A}\|^2 &= \Big\| G \bigtimes_{i=1}^{d} \mathbf{W}_i - \widehat{G} \bigtimes_{i=1}^{d} \mathbf{W}_i \Big\|^2 \\
&= \|G - \widehat{G}\|^2 \\
&= \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} (g_{i_1 \ldots i_d} - \widehat{g}_{i_1 \ldots i_d})^2 \\
&= \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} g_{i_1 \ldots i_d}^2 - \sum_{i_1=1}^{r_1} \cdots \sum_{i_d=1}^{r_d} g_{i_1 \ldots i_d}^2 \\
&\leq \sum_{i_1=r_1+1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_d=1}^{n_d} g_{i_1 \ldots i_d}^2 + \sum_{i_1=1}^{n_1} \sum_{i_2=r_2+1}^{n_2} \sum_{i_3=1}^{n_3} \cdots \sum_{i_d=1}^{n_d} g_{i_1 \ldots i_d}^2 + \cdots \\
&\quad + \sum_{i_1=1}^{n_1} \cdots \sum_{i_{d-1}=1}^{n_{d-1}} \sum_{i_d=r_d+1}^{n_d} g_{i_1 \ldots i_d}^2 \\
&= \|G - G_{r_1}\|^2 + \|G - G_{r_2}\|^2 + \cdots + \|G - G_{r_d}\|^2 \\
&= \sum_{j_1=r_1+1}^{n_1} \big(\sigma_{j_1}^{(1)}\big)^2 + \cdots + \sum_{j_d=r_d+1}^{n_d} \big(\sigma_{j_d}^{(d)}\big)^2,
\end{aligned}$$

where $G_{r_i} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is a subtensor of G of multilinear rank $(n_1, \ldots, n_{i-1}, r_i, n_{i+1}, \ldots, n_d)$, which is obtained by truncating G in the i-th mode, i.e. by keeping the first r_i singular values and setting the remaining elements to zero. Here, we have used the fact that the entries $\widehat{g}_{i_1 \ldots i_d}$ are either zero, or, since G_{r_i} is built from G, they are equal to the ones with the same indices in G. In the last equality, we used the Frobenius norm given in (4.7). Further, we can pull together the above error estimate for the THOSVD with the best multilinear low-rank approximation. Let G^* be the best rank $(r_1, \ldots, r_d)$ approximation to G. From the matrix case we know that the truncated SVD yields the best low-rank approximation of $\mathbf{Mat}_i(G)$ with rank r_i in the i-th mode. Hence, retensorizing yields

$$\|G - G_{r_i}\|^2 \leq \|G - G^*\|^2.$$

Now, let A^* be the best approximation to A with core G^*. Since the above inequality holds for each mode $i = 1, \ldots, d$ and since the factor matrices $\mathbf{W}_i$ are orthonormal for

each mode $i = 1, \dots, d$, we conclude that

$$\begin{aligned}\|A - \widehat{A}\| &\leq \sqrt{\|G - G_{r_1}\|^2 + \cdots + \|G - G_{r_d}\|^2} \\ &\leq \sqrt{\|G - G^*\|^2 + \cdots + \|G - G^*\|^2} \\ &= \sqrt{d} \cdot \|G - G^*\| \\ &= \sqrt{d} \cdot \|A - A^*\|.\end{aligned}$$

Therefore, the THOSVD may not result in the best rank $(r_1, \dots, r_d)$ approximation, but it still gives a quasi optimal approximation tensor of low multilinear rank.

Substituting the appropriate elements in the core tensor with zero keeps the size of the core tensor and so it does not provide an advantage about the computational cost. Now, since the multilinear product of $\widehat{G}$ with the factor matrices results in zero-entries in those matrices, we do not lose information when decreasing the size of $\widehat{G}$ simply by dropping the zero-entries to become a tensor $C \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ of full rank and restricting the factor matrices $\mathbf{W}_i$ to matrices $\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i}$ by truncating the last $(n_i - r_i)$ columns for all $i = 1, \dots, d$. This basic idea leads us to an *approximation tensor* Y to A, which is elementwise given as

$$y_{k_1,\dots,k_d} = \sum_{j_1=1}^{r_1} \cdots \sum_{j_d=1}^{r_d} c_{j_1,\dots,j_d} u_{k_1,j_1} \cdots u_{k_d,j_d},$$

or in shortform

$$Y = C \times_1 \mathbf{U}_1 \times_2 \cdots \times_d \mathbf{U}_d, \tag{4.9}$$

or in matricized form

$$\mathbf{Mat}_i(Y) = \mathbf{U}_i \cdot \mathbf{Mat}_i(C) \cdot \left(\mathbf{U}_1^\top \otimes \cdots \otimes \mathbf{U}_{i-1}^\top \otimes \mathbf{U}_{i+1}^\top \otimes \cdots \otimes \mathbf{U}_d^\top\right), \tag{4.10}$$

where Y is of the same size as A, but it is of low multilinear rank $(r_1, \dots, r_d)$ and the $\mathbf{U}_i$ are orthonormal $n_i \times r_i$ matrices for all $i = 1, \dots, d$.

We imagine Tucker tensors in the form of (4.9) as depicted in the following figure.

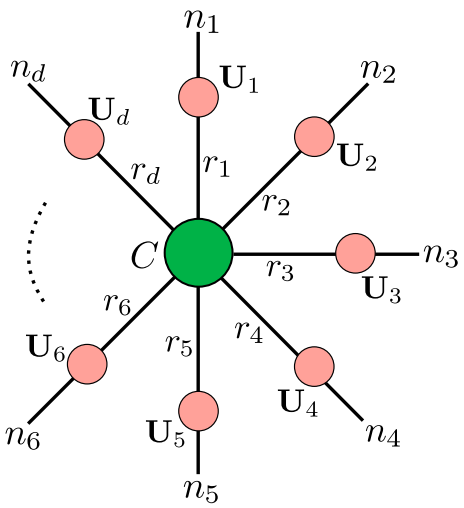

Figure 4.5: Illustration of a d-dimensional Tucker tensor.

Here, the midpoint represents the core tensor of size $r_1 \times \cdots \times r_d$ and the outer points stand for the factor matrices $\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i}$ with free indices $n_1, \ldots, n_d$. Now, the i-mode multiplication can be thought of as sticking the factor matrices to the core tensor, which is illustrated by the lines linking the core with each factor matrix.

Suppose $R = \max\{r_1, \ldots, r_d\}$. Then, storing the approximate tensor Y by storing the factors of its Tucker representation requires saving R^d entries of the core tensor C and $d \cdot (NR)$ entries of the factor matrices $\mathbf{U}_i$. In total, we can reduce the requirement of storage substantially, when assuming that $r_i \ll n_i$ for all $i = 1, \ldots, d$.

In fact, representation (4.9) is a truncated orthogonal Tucker decomposition, but instead of adding the descriptive terms "truncated" and "orthogonal", we will keep the notion *Tucker decomposition* for (4.9) for ease of notation. So from now on, we will think of this as the low-rank representation of the approximation tensor Y.

From the computational point of view, we can obtain the Tucker format (4.9), e.g. by Algorithm 5, which we call the *economic THOSVD* (eTHOSVD). This procedure is a straightforward extension of Algorithm 4 to the low-rank case. Since the HOSVD mainly consists of d SVDs, the natural extension for the high-dimensional case is to truncate the appearing matrices within the SVD in each dimension-step:

Algorithm 5: Tucker decomposition, eTHOSVD

Data: Tensor $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, $(r_1, \ldots, r_d)$
Result: Tucker tensor $C \times_1 \mathbf{U}_1 \times_2 \cdots \times_d \mathbf{U}_d \in \mathbb{R}^{n_1 \times \cdots \times n_d}$

```
begin
    for i = 1 to d do
        compute singular value decomposition Mat_i(A) = W_i Σ_i W̃_i^⊤
        truncate W_i(:, r_i) = U_i
    set C = A ×_1 U_1^⊤ ×_2 ··· ×_d U_d^⊤
    return C
    return U_1, ..., U_d
```

Besides the presented classic way of computing an approximation tensor of low multilinear rank, there are several different approaches. One of them is based on the THOSVD, but characterized by a *sequential* truncation strategy (sTHOSVD), see [VVM12]. It sequentially truncates the modes of the core tensor and computes a sequence of approximation tensors, whose multilinear ranks equal the desired dimension of the approximation space. A conceptually different method, based on an alternating least squares approach, is called higher order orthogonal iteration and was presented in [DLDMV00b].

4.2 The nested Tucker integrator

Turning to the main part of this chapter, we will consider time-dependent tensors and tensor differential equations. Since the factor matrices $\mathbf{U}_i$ are orthonormal in each mode i, we will consider them as being orthonormal basis matrices for the corresponding mode, i.e. $\mathbf{U}_i(t)$ is a basis matrix for $\mathbb{R}^{n_i \times r_i}$ for all times $t_0 \leq t \leq T$.

To find a low-rank approximation for (4.1) in the case of Tucker tensors of general dimension d, we will now extend the matrix projector-splitting integrator described in Section 2.1 to tensors. This newly derived integrator is first published in [LVW18], but here, we will present the integration method in much more detail. To this end, we will transfer a Tucker tensor into a matrix by considering its matricization $\mathbf{Mat}_i(Y)$ as given in (4.10). In a nutshell, the new Tucker integrator consists of successively applying the two-dimensional projector-splitting integrator, but with inexact solution in the third substep, which again is handled by a successive approximation. The first two substeps of the matrix integrator are applied and the corresponding matrices $\mathbf{K}$ and $\mathbf{S}$ are updated directly. The third substep is not solved directly, but by a low-rank approximation in the next mode. This nested procedure characterizes the integration method and that is the reason why we name it the *nested Tucker integrator*. The idea is not to consider the full tensor Y that in its matricized form is partly updated after having solved the first two steps, but to drop the already updated basis matrix $\mathbf{U}$ within the matricization of Y and only deal with the remaining factors of the Tucker representation of Y that are not updated yet. Leaving the already updated factors out leads to a beneficial side effect of the nested Tucker integrator: it reduces the complexity of the subproblems in each mode.

We begin by matricizing the *tensor differential equation* (4.1), viz.

$$\dot{A}(t) = F(t, A(t)), \qquad A(t_0) = A^0,$$

in mode 1, where we obtain the *matrix differential equation*

$$\mathbf{Mat}_1\big(\dot{A}(t)\big) = \mathbf{Mat}_1\big(F(t, A(t))\big), \qquad \mathbf{Mat}_1\big(A(t_0)\big) = \mathbf{Mat}_1\big(A^0\big). \tag{4.11}$$

This will allow us to apply the matrix projector-splitting integrator to this matrix ODE (4.11). The initial value $\mathbf{Mat}_1\big(A^0\big)$ is approximated by the matricization $\mathbf{Mat}_1\big(Y^0\big)$ of a low-rank tensor $Y^0 \in \mathcal{M}$. Since Y^0 is assumed to have multilinear rank $(r_1, \ldots, r_d)$, it satisfies a decomposition

$$Y^0 = C^0 \bigtimes_{i=1}^{d} \mathbf{U}_i^0,$$

with $C^0 \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ and $\mathbf{U}_i^0 \in \mathbb{R}^{n_i \times r_i}$. For clarity of notation, let us denote the initial value as $Y_1^0 := Y^0$ and the corresponding core tensor as $C_1^0 := C^0$. By performing a QR decomposition

$$\mathbf{Mat}_1(C_1^0)^\top = \mathbf{Q}_1^0 \, \mathbf{S}_1^{0,\top} \in \mathbb{R}^{r_2 \cdots r_d \times r_1},$$

where $\mathbf{Q}_1^0 \in \mathbb{R}^{r_2 \cdots r_d \times r_1}$ and $\mathbf{S}_1^0 \in \mathbb{R}^{r_1 \times r_1}$, and denoting

$$\mathbf{V}_1^{0,\top} = \mathbf{Q}_1^{0,\top} \bigotimes_{i=2}^{d} \mathbf{U}_i^{0,\top} \in \mathbb{R}^{r_1 \times n_2 \cdots n_d}, \tag{4.12}$$

we obtain the necessary SVD-like representation of the initial value of (4.11) as

$$\mathbf{Mat}_1(Y_1^0) = \mathbf{U}_1^0 \, \mathbf{Mat}_1(C_1^0) \bigotimes_{i=2}^{d} \mathbf{U}_i^{0,\top} = \mathbf{U}_1^0 \, \mathbf{S}_1^0 \, \mathbf{V}_1^{0,\top}. \tag{4.13}$$

Note that the matrix $\mathbf{S}_1^0$, which comes from the QR decomposition of the transposed 1-mode matricization of the core C^0, is not required to be diagonal. Further, in order to get the integrator started, a central idea is to assemble all basis matrices $\mathbf{U}_i^0$ with $i \neq 1$ in the matrix $\mathbf{V}_1^0$ as is done in (4.12) by using the Kronecker product. After having transformed the tensor setting into the matrix setting, we are now in the situation to apply the matrix projector-splitting integrator to (4.11) with initial value (4.13):

1. **K-step**: Update $\mathbf{U}_1^0 \to \mathbf{U}_1^1$, $\mathbf{S}_1^0 \to \widehat{\mathbf{S}}_1^1$ by solving *directly*

$$\begin{aligned} \dot{\mathbf{K}}_1(t) &= \mathbf{Mat}_1\big(F(t, \mathrm{Ten}_1(\mathbf{K}_1(t) \, \mathbf{V}_1^{0,\top}))\big) \, \mathbf{V}_1^0, \\ \mathbf{K}_1(t_0) &= \mathbf{U}_1^0 \, \mathbf{S}_1^0 \end{aligned}$$

   and performing a QR decomposition $\mathbf{K}_1(t_1) = \mathbf{U}_1^1 \, \widehat{\mathbf{S}}_1^1$

2. **S-step**: Update $\widehat{\mathbf{S}}_1^1 \to \widetilde{\mathbf{S}}_1^0$ by solving *directly*

$$\begin{aligned} \dot{\mathbf{S}}_1(t) &= -\,\mathbf{U}_1^{1,\top} \, \mathbf{Mat}_1\big(F(t, \mathrm{Ten}_1(\mathbf{U}_1^1 \, \mathbf{S}_1(t) \, \mathbf{V}_1^{0,\top}))\big) \, \mathbf{V}_1^0, \\ \mathbf{S}_1(t_0) &= \widehat{\mathbf{S}}_1^1 \end{aligned}$$

3. **L-step**: Update $\mathbf{V}_1^0 \to \mathbf{V}_1^1$, $\widetilde{\mathbf{S}}_1^0 \to \mathbf{S}_1^1$ by solving *approximately*

$$\begin{aligned} \dot{\mathbf{L}}_1(t)^\top &= \mathbf{U}_1^{1,\top} \, \mathbf{Mat}_1\left(F(t, \mathrm{Ten}_1(\mathbf{U}_1^1 \, \mathbf{L}_1(t)^\top))\right), \\ \mathbf{L}_1(t_0)^\top &= \widetilde{\mathbf{S}}_1^0 \, \mathbf{V}_1^{0,\top} \end{aligned} \tag{4.14}$$

   and performing a QR decomposition $\mathbf{L}_1(t_1) = \mathbf{V}_1^1 \, \mathbf{S}_1^{1,\top}$.

Note that computing the first two substeps *directly* means that we can directly apply an integration method in order to solve the differential equations for $\mathbf{K}_1$ and $\mathbf{S}_1$ without any further modifications of the ODEs. It is meant as the opposite to solving a differential equation *approximately*, e.g. by a low-rank approximation.

The **K**- and **S**-steps can be calculated as in the matrix case, but applied to (4.11). However, we do not solve the matrix differential equation in the **L**-step directly since it is defined for a prohibitively large $\mathbf{L}_1$. More importantly, it would also not lead to an approximation for $Y(t_1)$ of multilinear rank $(r_1, \ldots, r_d)$ since the exact **L**-step above only reduces the rank of the first mode. Instead, we perform a low-rank approximation for (4.14) by applying the matrix projector-splitting integrator again to a reshaped version of it.

Defining

$$Y_2(t) = \mathrm{Ten}_1(\mathbf{L}_1(t)^\top) \in \mathbb{R}^{r_1 \times n_2 \times \cdots \times n_d},$$

we first retensorize (4.14) as

$$\dot{Y}_2(t) = F(t, Y_2(t) \times_1 \mathbf{U}_1^1) \times_1 \mathbf{U}_1^{1,\top}, \qquad Y_2(t_0) = \mathrm{Ten}_1(\mathbf{L}_1^{0,\top}). \tag{4.15}$$

Observe that $Y_2(t)$ is the tensorized matrix $\mathbf{L}_1(t)^\top$, i.e. $Y_2(t)$ does not represent the partly updated full tensor $Y(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, since it does not contain the basis matrix $\mathbf{U}_1$. Therefore, $Y_2(t)$ is usually of significantly smaller size than $Y(t)$ since typically $r_1 \ll n_1$ and so the complexity of the problem is reduced, i.e. the differential equations that need to be solved are of smaller size, which is beneficial with regard to computational time and memory. Next, we unfold (4.15) in the second mode. For simplicity of notation, we denote this 2-mode unfolding of Y_2 by $\mathbf{Y}_{[2]} := \mathbf{Mat}_2(Y_2) \in \mathbb{R}^{n_2 \times r_1 n_3 \cdots n_d}$. This gives the *matrix* differential equation

$$\dot{\mathbf{Y}}_{[2]}(t) = \mathbf{Mat}_2\Big(F\big(t, \mathrm{Ten}_2(\mathbf{Y}_{[2]}(t)) \times_1 \mathbf{U}_1^1\big) \times_1 \mathbf{U}_1^{1,\top}\Big) \tag{4.16}$$

and, using (4.14) and (4.12), the initial value

$$\mathbf{Y}_{[2]}(t_0) = \mathbf{Mat}_2\left(\mathrm{Ten}_1(\mathbf{L}_1^{0,\top})\right) = \mathbf{Mat}_2\Big(\mathrm{Ten}_1\big(\widetilde{\mathbf{S}}_1^0\, \mathbf{Q}_1^{0,\top} \bigotimes_{i=2}^{d} \mathbf{U}_i^{0,\top}\big)\Big).$$

Defining $C_2^0 = \mathrm{Ten}_1\big(\widetilde{\mathbf{S}}_1^0\, \mathbf{Q}_1^{0,\top}\big) \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ and $\mathbf{C}_{[2]}^0 = \mathbf{Mat}_2(C_2^0)$, we also have

$$\mathbf{Y}_{[2]}(t_0) = \mathbf{Mat}_2\Big(C_2^0 \bigtimes_{i=2}^{d} \mathbf{U}_i^0\Big) = \mathbf{Mat}_2\Big(C_2^0 \times_1 \mathbf{I}_{r_1} \bigtimes_{i=2}^{d} \mathbf{U}_i^0\Big) = \mathbf{U}_2^0\, \mathbf{C}_{[2]}^0 \Big(\mathbf{I}_{r_1} \otimes \big(\bigotimes_{i=3}^{d} \mathbf{U}_i^{0,\top}\big)\Big).$$

In order to be able to again apply the matrix projector-splitting integrator, we have to determine the SVD-like representation of $\mathbf{Y}_{[2]}(t_0)$ as before. To this end, we compute the QR factorization $\mathbf{C}_{[2]}^{0,\top} = \mathbf{Q}_2^0\, \mathbf{S}_2^{0,\top}$. We then obtain the desired result as

$$\mathbf{Y}_{[2]}(t_0) = \mathbf{U}_2^0\, \mathbf{S}_2^0\, \mathbf{Q}_2^{0,\top} \Big(\mathbf{I}_{r_1} \otimes \big(\bigotimes_{i=3}^{d} \mathbf{U}_i^{0,\top}\big)\Big) = \mathbf{U}_2^0\, \mathbf{S}_2^0\, \mathbf{V}_2^{0,\top},$$

with $\mathbf{V}_2^{0,\top} = \mathbf{Q}_2^{0,\top} \Big(\mathbf{I}_{r_1} \otimes \big(\bigotimes_{i=3}^{d} \mathbf{U}_i^{0,\top}\big)\Big) \in \mathbb{R}^{r_2 \times r_1 n_3 \cdots n_d}$.

Now that we have set up the matrix problem again, we can apply the matrix projector-splitting integrator to (4.16):

1. **K-step**: Update $\mathbf{U}_2^0 \to \mathbf{U}_2^1$, $\mathbf{S}_2^0 \to \widehat{\mathbf{S}}_2^1$ by solving *directly*

$$\begin{aligned} \dot{\mathbf{K}}_2(t) &= \mathbf{Mat}_2\Big(F\big(t, \mathrm{Ten}_2(\mathbf{K}_2(t)\, \mathbf{V}_2^{0,\top}) \times_1 \mathbf{U}_1^1\big) \times_1 \mathbf{U}_1^{1,\top}\Big)\, \mathbf{V}_2^0, \\ \mathbf{K}_2(t_0) &= \mathbf{U}_2^0\, \mathbf{S}_2^0 \end{aligned}$$

   and performing a QR decomposition $\mathbf{K}_2(t_1) = \mathbf{U}_2^1\, \widehat{\mathbf{S}}_2^1$

2. **S-step**: Update $\widehat{\mathbf{S}}_2^1 \to \widetilde{\mathbf{S}}_2^0$ by solving *directly*

$$\begin{aligned} \dot{\mathbf{S}}_2(t) &= -\,\mathbf{U}_2^{1,\top}\, \mathbf{Mat}_2\Big(F\big(t, \mathrm{Ten}_2(\mathbf{U}_2^1\, \mathbf{S}_2(t)\, \mathbf{V}_2^{0,\top}) \times_1 \mathbf{U}_1^1\big) \times_1 \mathbf{U}_1^{1,\top}\Big)\, \mathbf{V}_2^0, \\ \mathbf{S}_2(t_0) &= \widehat{\mathbf{S}}_2^1 \end{aligned}$$

3. **L-step**: Update $\mathbf{V}_2^0 \to \mathbf{V}_2^1$, $\widetilde{\mathbf{S}}_2^0 \to \mathbf{S}_2^1$ by solving *approximately*

$$\begin{aligned}\dot{\mathbf{L}}_2(t)^\top &= \mathbf{U}_2^{1,\top}\,\mathbf{Mat}_2\Big(F\big(t, \mathrm{Ten}_2(\mathbf{U}_2^1\,\mathbf{L}_2(t)^\top)\times_1 \mathbf{U}_1^1\big)\times_1 \mathbf{U}_1^{1,\top}\Big),\\ \mathbf{L}_2(t_0)^\top &= \widetilde{\mathbf{S}}_2^0\,\mathbf{V}_2^{0,\top}\end{aligned}$$

and performing a QR decomposition $\mathbf{L}_2(t_1) = \mathbf{V}_2^1\,\mathbf{S}_2^{1,\top}$.

We continue recursively with solving the **K**- and the **S**-step as in the matrix case and solving the **L**-step approximately in each iteration step of the integrator. Generalizing the pattern for modes 1 and 2 from above to a general mode i requires us to find $Y_i(t) \in \mathbb{R}^{r_1\times\cdots\times r_{i-1}\times n_i\times\cdots\times n_d}$ that satisfies the ODE

$$\dot{Y}_i(t) = F\Big(t, Y_i(t)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1\Big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}, \qquad Y_i(t_0) = \mathrm{Ten}_{i-1}\big(\mathbf{L}_{i-1}^{0,\top}\big). \tag{4.17}$$

The **K**- and **S**-steps for mode $(i-1)$ calculate, in particular, the matrix $\widetilde{\mathbf{S}}_{i-1}^0$. The orthogonal matrix $\mathbf{Q}_{i-1}^0$ is obtained from the updated core tensor C_{i-1}^0 in the sense that

$$\mathbf{Mat}_{i-1}(C_{i-1}^0)^\top = \mathbf{Q}_{i-1}^0\,\mathbf{S}_{i-1}^{0,\top} \in \mathbb{R}^{r_1\cdots r_{i-2}r_i\cdots r_d\times r_{i-1}}.$$

This implies that the initial value in the above ODE is available in the matricized $(i-1)$ mode as

$$\mathbf{L}_{i-1}^{0,\top} = \widetilde{\mathbf{S}}_{i-1}^0\,\mathbf{Q}_{i-1}^{0,\top}\Big((\bigotimes_{k=1}^{i-2}\mathbf{I}_{r_k})\otimes(\bigotimes_{k=i}^{d}\mathbf{U}_k^{0,\top})\Big) \in \mathbb{R}^{r_{i-1}\times r_1\cdots r_{i-2}n_i\cdots n_d}.$$

To obtain a suitable matrix version of (4.17), we matricize it in mode i and define $\mathbf{Y}_{[i]} = \mathbf{Mat}_i(Y_i) \in \mathbb{R}^{n_i\times r_1\cdots r_{i-1}n_{i+1}\cdots n_d}$. This gives

$$\begin{aligned}\dot{\mathbf{Y}}_{[i]}(t) &= \mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i\big(\mathbf{Y}_{[i]}(t)\big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1\Big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big),\\ \mathbf{Y}_{[i]}(t_0) &= \mathbf{Mat}_i\Big(\mathrm{Ten}_{i-1}\Big(\widetilde{\mathbf{S}}_{i-1}^0\,\mathbf{Q}_{i-1}^{0,\top}\Big((\bigotimes_{k=1}^{i-2}\mathbf{I}_{r_k})\otimes(\bigotimes_{k=i}^{d}\mathbf{U}_k^{0,\top})\Big)\Big)\Big)\\ &= \mathbf{Mat}_i\Big(C_i^0\bigtimes_{k=1}^{i-1}\mathbf{I}_{r_k}\bigtimes_{k=i}^{d}\mathbf{U}_k^0\Big),\end{aligned} \tag{4.18}$$

with $C_i^0 = \mathrm{Ten}_{i-1}\big(\widetilde{\mathbf{S}}_{i-1}^0\,\mathbf{Q}_{i-1}^{0,\top}\big) \in \mathbb{R}^{r_1\times\cdots\times r_d}$. In order to obtain a SVD-like decomposition of the initial value $\mathbf{Y}_{[i]}(t_0)$, we perform a QR decomposition

$$\mathbf{Mat}_i(C_i^0)^\top = \mathbf{Q}_i^0\,\mathbf{S}_i^{0,\top} \in \mathbb{R}^{r_1\cdots r_{i-1}r_{i+1}\cdots r_d\times r_i}$$

and set

$$\mathbf{V}_i^{0,\top} = \mathbf{Q}_i^{0,\top}\Big((\bigotimes_{k=1}^{i-1}\mathbf{I}_{r_k})\otimes(\bigotimes_{k=i+1}^{d}\mathbf{U}_k^{0,\top})\Big) \in \mathbb{R}^{r_i\times r_1\cdots r_{i-1}n_{i+1}\cdots n_d}. \tag{4.19}$$

In this way, we indeed obtain

$$\mathbf{Y}_{[i]}(t_0) = \mathbf{Mat}_i\Big(C_i^0\bigtimes_{k=1}^{i-1}\mathbf{I}_{r_k}\bigtimes_{k=i}^{d}\mathbf{U}_k^0\Big) = \mathbf{U}_i^0\,\mathbf{S}_i^0\,\mathbf{V}_i^{0,\top}$$

and therefore we can apply the **K**- and **S**-steps of the matrix projector-splitting integrator to (4.18). The **L**-step consists of recursively solving

$$\begin{aligned}\dot{\mathbf{L}}_i(t)^\top &= \mathbf{U}_i^{1,\top}\,\mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i\big(\mathbf{U}_i^1\,\mathbf{L}_i(t)^\top\big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1\Big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big),\\ \mathbf{L}_i(t_0)^\top &= \mathbf{L}_i^{0,\top} = \widetilde{\mathbf{S}}_i^0\,\mathbf{V}_i^{0,\top},\end{aligned} \tag{4.20}$$

with the scheme we just explained. We continue this procedure until mode $(d-1)$. Having reached mode d, the recursion changes its line of action in the third substep. In the last mode, we have to determine an approximation tensor $Y_d(t)$ of size $r_1 \times \cdots \times r_{d-1} \times n_d$ that solves the *tensor* ODE

$$\dot{Y}_d(t) = F\Big(t, Y_d(t)\bigtimes_{k=1}^{d-1}\mathbf{U}_k^1\Big)\bigtimes_{k=1}^{d-1}\mathbf{U}_k^{1,\top}, \qquad Y_d(t_0) = \mathrm{Ten}_{d-1}\big(\mathbf{L}_{d-1}^{0,\top}\big). \tag{4.21}$$

Now, since the initial value in the $(d-1)$-mode matricization is given by

$$\mathbf{L}_{d-1}^{0,\top} = \widetilde{\mathbf{S}}_{d-1}^0\,\mathbf{V}_{d-1}^{0,\top} = \widetilde{\mathbf{S}}_{d-1}^0\,\mathbf{Q}_{d-1}^{0,\top}\Big(\bigotimes_{k=1}^{d-2}\mathbf{I}_{r_k}\otimes\mathbf{U}_d^{0,\top}\Big),$$

we pass it to the d-mode matricized form by first retensorizing and then matricizing, i.e.,

$$\begin{aligned}\mathbf{Y}_{[d]}^{0,\top} &= \mathbf{Mat}_d\Big(\mathrm{Ten}_{d-1}\Big(\widetilde{\mathbf{S}}_{d-1}^0\,\mathbf{Q}_{d-1}^{0,\top}\Big(\bigotimes_{k=1}^{d-2}\mathbf{I}_{r_k}\otimes\mathbf{U}_d^{0,\top}\Big)\Big)\Big)\\ &= \mathbf{Mat}_d\Big(C_d^0\bigtimes_{k=1}^{d-1}\mathbf{I}_{r_k}\times_d\mathbf{U}_d^0\Big)\\ &= \mathbf{U}_d^0\,\mathbf{S}_d^0\,\mathbf{Q}_d^{0,\top}\Big(\bigotimes_{k=1}^{d-1}\mathbf{I}_{r_k}\Big)\\ &= \mathbf{U}_d^0\,\mathbf{S}_d^0\,\mathbf{V}_d^{0,\top},\end{aligned}$$

where $C_d^0 = \mathrm{Ten}_{d-1}(\widetilde{\mathbf{S}}_{d-1}^0\,\mathbf{Q}_{d-1}^{0,\top})$, $\mathbf{Mat}_d(C_d^0)^\top = \mathbf{Q}_d^0\,\mathbf{S}_d^{0,\top}$ and setting

$$\mathbf{V}_d^{0,\top} = \mathbf{Q}_d^{0,\top}\Big(\bigotimes_{k=1}^{d-1}\mathbf{I}_{r_k}\Big). \tag{4.22}$$

Compared to the previous mode-steps, an important difference can already be seen at this stage. We observe that the matrix $\mathbf{V}_d^{0,\top}$ does not contain a basis matrix at time t_0 in the Kronecker product. This is an essential observation, which involves a direct computation of the third substep within the matrix projector-splitting integrator, as will be seen in the following:

1. **K-step**: Update $\mathbf{U}_d^0 \to \mathbf{U}_d^1$, $\mathbf{S}_d^0 \to \widehat{\mathbf{S}}_d^1$ by solving *directly*

$$\begin{aligned}\dot{\mathbf{K}}_d(t) &= \mathbf{Mat}_d\big(F(t, \mathrm{Ten}_d(\mathbf{K}_d(t)\,\mathbf{V}_d(t)^\top))\big)\,\mathbf{V}_d^0,\\ \mathbf{K}_d(t_0) &= \mathbf{U}_d^0\,\mathbf{S}_d^0\end{aligned}$$

and performing a QR decomposition $\mathbf{K}_d(t_1) = \mathbf{U}_d^1\,\widehat{\mathbf{S}}_d^1$

2. **S-step**: Update $\widehat{\mathbf{S}}_d^1 \to \widetilde{\mathbf{S}}_d^0$ by solving *directly*

$$\begin{aligned}\dot{\mathbf{S}}_d(t) &= -\,\mathbf{U}_d^{1,\top}\,\mathbf{Mat}_d\big(F(t, \mathrm{Ten}_d(\mathbf{U}_d^1\,\mathbf{S}_d(t)\,\mathbf{V}_d^{0,\top}))\big)\,\mathbf{V}_d^0,\\ \mathbf{S}_d(t_0) &= \widehat{\mathbf{S}}_d^1\end{aligned}$$

3. **L-step**: Update $\mathbf{V}_d^0 \to \mathbf{V}_d^1$, $\widetilde{\mathbf{S}}_d^0 \to \mathbf{S}_d^1$ by solving *directly*

$$\begin{aligned}\dot{\mathbf{L}}_d(t)^\top &= \mathbf{U}_d^{1,\top}\,\mathbf{Mat}_d\Big(F\Big(t, \mathrm{Ten}_d\big(\mathbf{U}_d^1\,\mathbf{L}_d^\top\big)\bigtimes_{i=1}^{d-1}\mathbf{U}_i^1\Big)\bigtimes_{i=1}^{d-1}\mathbf{U}_i^{1,\top}\Big),\\ \mathbf{L}_d(t_0)^\top &= \mathbf{L}_d^{0,\top} = \widetilde{\mathbf{S}}_d^0\,\mathbf{V}_d^{0,\top}\end{aligned} \tag{4.23}$$

and performing a QR decomposition $\mathbf{L}_d(t_1) = \mathbf{V}_d^1\,\mathbf{S}_d^{1,\top}$.

In (4.22) we observe that the matrix $\mathbf{V}_d^0$ solely consists of the orthogonal part $\mathbf{Q}_d^0$ of the core tensor. Therefore, we have

$$\mathbf{L}_d^{0,\top} = \Big(\widetilde{\mathbf{S}}_d^0\,\mathbf{Q}_d^{0,\top}\Big) \otimes \Big(\bigotimes_{i=1}^{d-1}\mathbf{I}_{r_i}\Big) = \mathbf{Mat}_d(C_d^0) \in \mathbb{R}^{r_d \times r_1 \cdots r_{d-1}},$$

which means that $\mathbf{L}_d(t)$ actually corresponds to the core $C(t)$ itself. Hence, solving (4.23) yields $\mathrm{Ten}_d(\mathbf{L}_d^{1,\top}) = C_d^1 =: C^1$, which is an approximation to $C(t)$ after one time step. We illustrate the scheme of the nested Tucker integrator in Figure 4.6.

The nested Tucker integrator presented above operates on tensors $Y_i(t)$ that consecutively get smaller for $i = 1, 2, \ldots, d$. However, we can also interpret it as computing an approximation Y^1 for the Tucker tensor $Y(t_1) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ in (4.2). In particular, we have

$$Y^1 = \mathrm{Ten}_1\big(\mathbf{U}_1^1\,\mathbf{L}_1(t_1)^\top\big) = \mathrm{Ten}_1\left(\mathbf{U}_1^1\,\mathbf{Mat}_1(Y_2(t_1))\right) = Y_2(t_1) \times_1 \mathbf{U}_1^1,$$

with $\mathbf{L}_1(t_1)$ the approximate solution of (4.14) obtained using $Y_2(t_1)$ in (4.15). In turn, $Y_2(t_1)$ is solved similarly using $Y_3(t_1)$:

$$Y_2(t_1) = \mathrm{Ten}_2\big(\mathbf{U}_2^1\,\mathbf{L}_2(t_1)^\top\big) = Y_3(t_1) \times_2 \mathbf{U}_2^1\,.$$

Hence, continuing recursively for all modes, we obtain the desired approximation tensor after one time step at $t_1 = t_0 + h$ in Tucker format, which is determined as

$$\begin{aligned}Y^1 &= \mathrm{Ten}_1(\mathbf{U}_1^1\,\mathbf{L}_1^{1,\top}) = \mathrm{Ten}_1\big(\mathbf{U}_1^1\,\mathbf{Mat}_1(Y_2^1)\big) = Y_2^1 \times_1 \mathbf{U}_1^1\\ &= \mathrm{Ten}_2(\mathbf{U}_2^1\,\mathbf{L}_2^{1,\top}) \times_1 \mathbf{U}_1^1 = \mathrm{Ten}_2\big(\mathbf{U}_2^1\,\mathbf{Mat}_2(Y_3^1)\big) \times_1 \mathbf{U}_1^1 = Y_3^1 \times_2 \mathbf{U}_2^1 \times_1 \mathbf{U}_1^1\\ &\;\;\vdots\\ &= \mathrm{Ten}_i(\mathbf{U}_i^1\,\mathbf{L}_i^{1,\top})\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1 = \mathrm{Ten}_i\big(\mathbf{U}_i^1\,\mathbf{Mat}_i(Y_{i+1}^1)\big)\bigtimes_{k=1}^{i-1} = Y_{i+1}^1\bigtimes_{k=1}^{i}\mathbf{U}_k^1\\ &\;\;\vdots\\ &= C^1\bigtimes_{k=1}^{d}\mathbf{U}_k^1 \approx Y(t_1) \approx A(t_1).\end{aligned}$$

If we now want to continue in time to approximate $Y(t_2)$ for $t_2 \geq t_1$, we apply the integration scheme again using Y^1 as initial value and analogously for the subsequent time steps.

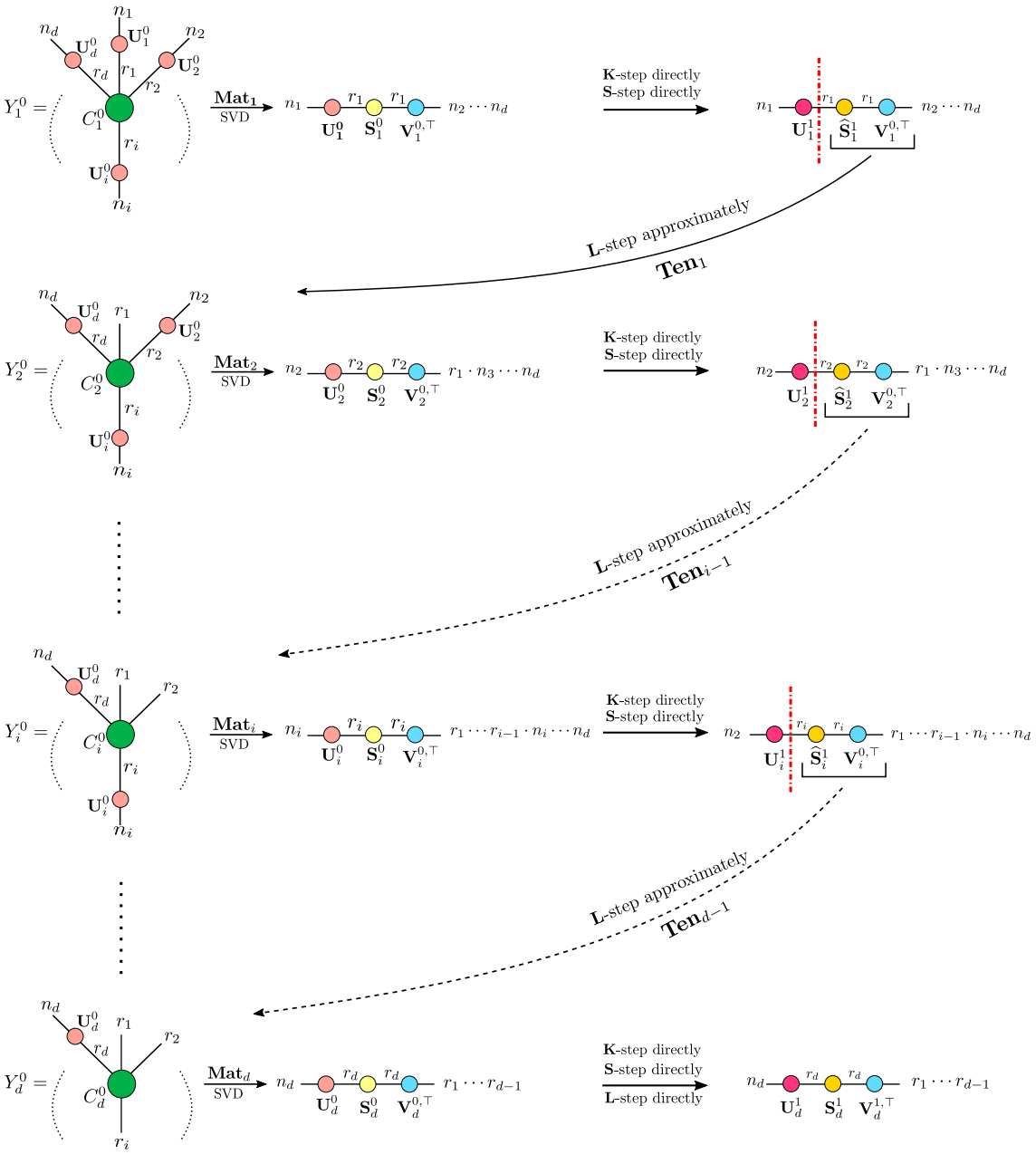

Figure 4.6: Illustration of the nested Tucker integrator.

4.3 Algorithmic description of the nested Tucker integrator

As we have seen, the nested Tucker integrator follows the scheme of recursively applying the matrix projector-splitting integrator described in Section 2.1 with solving the first

two steps directly, but performing a low-rank approximation for the third substep in each mode. So it simply updates the basis matrices $\mathbf{U}_i$ in the **K**-step and the auxiliary matrix $\mathbf{S}_i$ in the **S**-step for each mode. Then, quite remarkably, to start the computation for the approximate **L**-step, i.e. transfering the integration from one mode to the subsequent one, it suffices to only update the (intermediate) core tensor. Since

$$\mathbf{L}_i^\top = \widetilde{\mathbf{S}}_i \, \mathbf{V}_i^\top,$$

where $\mathbf{V}_i^\top$ contains all basis matrices $\mathbf{U}_i, \dots, \mathbf{U}_d$ that are not updated yet, we imagine to simply drop the already updated basis matrix $\mathbf{U}_{i-1}$, which is depicted in Figure 4.6 by the red dashed line. Leaving out the basis matrix in the current mode leads to reducing the size of the matrix differential equation that has to be solved for the next mode. The nestedness comes from the fact that the **L**-step is not calculated directly, but by applying the matrix projector-splitting integrator to the matricized situation in the subsequent mode for all modes $i = 1, \dots, d-1$. Hence this integration scheme is a nested matrix projector-splitting integrator for Tucker tensors, or in short, the nested Tucker integrator.

Algorithm 6: One time step of the nested Tucker integrator

Data: Tucker tensor $Y^0 = C^0 \bigtimes_{i=1}^d \mathbf{U}_i^0$, $F(t, Y)$, t_0, t_1
Result: Tucker tensor $Y^1 = C^1 \bigtimes_{i=1}^d \mathbf{U}_i^1$

1 **begin**
2 **for** $i = 1$ **to** d **do**
3 compute QR factorization $\mathbf{Mat}_i(C^0)^\top = \mathbf{Q}_i^0 \, \mathbf{S}_i^{0,\top}$
4 set $\mathbf{V}_i^{0,\top} = \mathbf{Mat}_i\Big(\mathrm{Ten}_i(\mathbf{Q}_i^{0,\top}) \bigtimes_{l=i+1}^{d} \mathbf{U}_l^0\Big)$
5 set $\mathbf{K}_i^0 = \mathbf{U}_i^0 \, \mathbf{S}_i^0$
6 set $\mathbf{Y}_{[i]}^+(t) = \mathbf{K}_i(t) \, \mathbf{V}_i^{0,\top}$
7 solve $\dot{\mathbf{K}}_i(t) = \mathbf{Mat}_i\Big(F\big(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^+) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big) \, \mathbf{V}_i^0$,
  with initial value $\mathbf{K}_i(t_0) = \mathbf{K}_i^0$ and return $\mathbf{K}_i^1 = \mathbf{K}_i(t_1)$
8 compute QR factorization $\mathbf{K}_i^1 = \mathbf{U}_i^1 \, \widehat{\mathbf{S}}_i^1$
9 set $\mathbf{Y}_{[i]}^-(t) = \mathbf{U}_i^1 \, \mathbf{S}_i(t) \, \mathbf{V}_i^{0,\top}$
10 solve $\dot{\mathbf{S}}_i(t) = -\,\mathbf{U}_i^{1,\top} \, \mathbf{Mat}_i\Big(F\big(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^-) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big) \, \mathbf{V}_i^0$,
  with initial value $\mathbf{S}_i(t_0) = \widehat{\mathbf{S}}_i^1$ and return $\widetilde{\mathbf{S}}_i^0 = \mathbf{S}_i(t_1)$
11 set $C^0 = \mathrm{Ten}_i(\widetilde{\mathbf{S}}_i^0 \, \mathbf{Q}_i^{0,\top})$
12 set $\mathbf{L}^{0,\top} = \mathbf{Mat}_d(C^0)$
13 solve $\dot{\mathbf{L}}(t)^\top = \mathbf{U}_d^{1,\top} \, \mathbf{Mat}_d\Big(F\big(t, \mathrm{Ten}_d(\mathbf{U}_d^1 \, \mathbf{L}(t)^\top) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^1\big) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^{1,\top}\Big)$,
  with initial value $\mathbf{L}(t_0)^\top = \mathbf{L}^{0,\top}$ and return $\mathbf{L}^{1,\top} = \mathbf{L}(t_1)^\top$
14 set $C^1 = \mathrm{Ten}_d(\mathbf{L}^{1,\top})$
15 set $Y^1 = C^1 \bigtimes_{i=1}^{d} \mathbf{U}_i^1$

For computational efficiency, we have written the operations using multilinear products. For example, line 4 is equivalent to (4.19).

From the implementational point of view, the differential equations for $\mathbf{K}, \mathbf{S}$ and $\mathbf{L}$ that need to be solved during the integration scheme, can be solved approximately, e.g., by a classical 4th-order Runge–Kutta method without substeps. On the other hand, if $F(t, Y)$ is solution-independent and solely given by a tensor $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, we can determine a closed-form solution for those differential equations.

4.4 An exactness property of the nested Tucker integrator

In this section, we give a proof of the exactness of the nested Tucker integrator, which is first performed in [LVW18, Theorem 4.1].

Suppose that $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is given explicitly, hence, we formally have $F(t, Y) = \dot{A}(t)$ in (4.1) and (4.2). In addition, we assume that $A(t) \in \mathcal{M}$ for $t_0 \leq t \leq T$. Our aim in this section is to prove that Algorithm 6, the nested Tucker integrator, is exact in that case. In other words, Algorithm 6 solves the initial value problem (4.2) exactly even though it is a discrete time stepping method. As stated in Theorem 2.1, the projector-splitting integrator for matrices already has this property.

Since $A(t)$ is of multilinear rank $(r_1, \ldots, r_d)$ for all t, we can write its i-mode matricization as

$$\mathbf{Mat}_i(A(t)) = \mathbf{U}_i(t)\mathbf{S}_i(t)\widehat{\mathbf{V}}_i(t)^\top, \tag{4.24}$$

where $\mathbf{U}_i(t) \in \mathbb{R}^{n_i \times r_i}$ and $\widehat{\mathbf{V}}_i(t) \in \mathbb{R}^{n_1 \cdots n_{i-1} n_{i+1} \cdots n_d \times r_i}$ have orthonormal columns and $\mathbf{S}_i(t) \in \mathbb{R}^{r_i \times r_i}$ for all $i = 1, \ldots, d$. With this SVD-like representation we can state and prove the following exactness result.

Theorem 4.1 (Exactness of the nested Tucker integrator)**.**
Let $A(t)$ be of multilinear rank $(r_1, \ldots, r_d)$ for all $t \in (t_0, t_1)$ and let $Y(t_0) = A(t_0)$. Further, let $\mathbf{V}_i(t_1)^\top \mathbf{V}_i(t_0)$, where $\mathbf{V}_i(t)$ is defined in the course of the algorithm, see (4.28), be invertible for all $i = 2, \ldots, d$. Then, Algorithm 6 for $F(t, Y) = \dot{A}(t)$ reproduces the exact solution: $Y^1 = A(t_1)$.

Proof. Recall that the nested Tucker integrator in Algorithm 6 is designed to approximately solve the initial value subproblems

$$\dot{\mathbf{Y}}_{[i]}(t) = \mathbf{Mat}_i\Big(\dot{A}(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big), \qquad \mathbf{Y}_{[i]}(t_0) = \mathbf{Y}_{[i]}^0 = \mathbf{U}_i^0 \, \mathbf{S}_i^0 \, \mathbf{V}_i^{0,\top}, \tag{4.25}$$

where $\mathrm{Ten}_i(\mathbf{V}_i^{0,\top}) = \mathrm{Ten}_i(\mathbf{Q}_i^{0,\top}) \bigtimes_{l=i+1}^{d} \mathbf{U}_l^0 \in \mathbb{R}^{r_1 \times \cdots \times r_i \times n_{i+1} \times \cdots \times n_d}$ for each mode $i = 1, \ldots, d$. In addition, the tensorized result $Y_i^1 = \mathrm{Ten}_i(\mathbf{Y}_{[i]}(t_1))$ after one time step is in the low-rank manifold $\mathcal{M}_i := \{Y_i \in \mathbb{R}^{r_1 \times \cdots \times r_{i-1} \times n_i \times \cdots \times n_d} : \operatorname{rank} \mathbf{Mat}_i(Y_i) = r_i \in \mathbb{N}\}$.

Since the nested Tucker integrator is based on the matrix projector-splitting integrator, which exhibits the favorable exactness result, we want to make use of this property for the two-dimensional case. There, an essential assumption is made on the matrix that needs to be approximated: it has to be of low rank. Hence, in order to benefit from this result, we have to show that the initial values in each modal step within the integrator satisfy this

property. So first we show by induction that the initial value for (4.25) can be written in terms of $A(t_0)$:

$$\mathbf{Y}_{[i]}(t_0) = \mathbf{Mat}_i\Big(A(t_0)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big). \tag{4.26}$$

This ensures that $\mathbf{Y}_{[i]}^0$ has rank r_i. Suppose that this has been shown for $\mathbf{Y}_{[1]}^0,\dots,\mathbf{Y}_{[i-1]}^0$. Now, due to the nestedness of the integrator, we have $Y_i(t_0) = \mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,\top})$. So in order to assure that $Y_i(t_0)$ is of low rank, we have to take a closer look on $\mathbf{L}_{i-1}^{0,\top}$. With the abbreviation $\Delta A = A(t_1) - A(t_0)$, it is

$$\begin{aligned}
\mathbf{U}_{i-1}^1\,\mathbf{L}_{i-1}^{0,\top} &= \mathbf{U}_{i-1}^1\,\widehat{\mathbf{S}}_{i-1}^1\,\mathbf{V}_{i-1}^{0,\top} \;-\; \mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,\top}\,\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,\top}\\
&= \mathbf{U}_{i-1}^0\,\mathbf{S}_{i-1}^0\,\mathbf{V}_{i-1}^{0,\top} + \mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,\top}\\
&\quad - \mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,\top}\,\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,\top}\\
&= \mathbf{Y}_{[i-1]}^0 + \big(\mathbf{I} - \mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,\top}\big)\Big(\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,\top}\Big)\\
&= \mathbf{Mat}_{i-1}\big(\mathrm{Ten}_{i-2}(\mathbf{L}_{i-2}^{0,\top})\big)\\
&\quad + \big(\mathbf{I} - \mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,\top}\big)\Big(\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,\top}\Big)\\
&= \mathbf{Mat}_{i-1}\Big(A(t_0)\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big)\\
&\quad + \big(\mathbf{I} - \mathbf{U}_{i-1}^1\,\mathbf{U}_{i-1}^{1,\top}\big)\Big(\mathbf{Mat}_{i-1}\Big(\Delta A\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_{i-1}^0\,\mathbf{V}_{i-1}^{0,\top}\Big),
\end{aligned}$$

where the last equality holds by induction hypothesis. Multiplying both sides by $\mathbf{U}_{i-1}^{1,\top}$ from the left yields

$$\mathbf{L}_{i-1}^{0,\top} = \mathbf{U}_{i-1}^{1,\top}\,\mathbf{Mat}_{i-1}\Big(A(t_0)\bigtimes_{k=1}^{i-2}\mathbf{U}_k^{1,\top}\Big) = \mathbf{Mat}_{i-1}\Big(A(t_0)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big). \tag{4.27}$$

By retensorizing and matricizing in the i-th mode, we obtain

$$\mathbf{Y}_{[i]}(t_0) = \mathbf{Mat}_i\big(\mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,\top})\big) = \mathbf{Mat}_i\Big(A(t_0)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big).$$

With this initial value, the exact solution of (4.25) has rank r_i as well, since $A(t)$ is assumed to have multilinear rank $(r_1,\dots,r_d)$ for all $t \in (t_0,t_1)$:

$$\begin{aligned}
\mathbf{Y}_{[i]}(t) &= \mathbf{Y}_{[i]}(t_0) + \int_{t_0}^{t}\dot{\mathbf{Y}}_{[i]}(s)ds\\
&= \mathbf{Mat}_i\Big(A(t_0)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big) + \mathbf{Mat}_i\Big(\big(A(t) - A(t_0)\big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big)\\
&= \mathbf{Mat}_i\Big(A(t)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big)\\
&= \mathbf{U}_i(t)\mathbf{S}_i(t)\,\mathbf{V}_i(t)^\top,
\end{aligned}$$

where we use the decomposition (4.24) and set

$$\mathbf{V}_i(t) = (\mathbf{U}_1^{1,\top} \otimes \cdots \otimes \mathbf{U}_{i-1}^{1,\top} \otimes \mathbf{I}_{r_{i+1}} \otimes \cdots \otimes \mathbf{I}_{r_d})\, \widehat{\mathbf{V}}_i(t). \tag{4.28}$$

To show the exactness of Algorithm 6, we first consider the d-mode unfolded subproblem. Here, the last substep of the nested Tucker integrator is the same as applying the matrix projector-splitting integrator to (4.25) with initial value (4.26) for $i = d$. Since the updated basis matrices $\mathbf{U}_k^1$ for $k = 1, \dots i-1$ are not time-dependent from the i-th integration step onwards, we observe by means of (4.28) that

$$\mathbf{V}_i(t_1)^\top \mathbf{V}_i(t_0) = \widehat{\mathbf{V}}_i(t_1)^\top (\mathbf{U}_1^1 \mathbf{U}_1^{1,\top} \otimes \cdots \otimes \mathbf{U}_{i-1}^1 \mathbf{U}_{i-1}^{1,\top} \otimes \mathbf{I}_{r_{i+1}} \otimes \cdots \otimes \mathbf{I}_{r_d})\, \widehat{\mathbf{V}}_i(t_0)$$

for all $i = 1, \dots, d$. Additionally, by assumption, $\mathbf{V}_i(t_1)^\top \mathbf{V}_i(t_0)$ is non-singular and so we conclude by Theorem 2.1 that the integrator is exact for the d-mode setting after one time step from t_0 to t_1:

$$\mathbf{Y}_{[d]}^1 = \mathbf{Mat}_d\Big(A(t_1) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^{1,\top}\Big).$$

We now show by induction for $i = d, \dots, 1$ that

$$\mathbf{Y}_{[i]}^1 = \mathbf{Mat}_i\Big(A(t_1) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big). \tag{4.29}$$

Suppose this has been shown for $\mathbf{Y}_{[d]}^1, \dots, \mathbf{Y}_{[i+1]}^1$. Since the initial value for the mode i subproblem is given by (4.26), the corresponding initial values for the three subproblems in this mode can be written in terms of $\mathbf{Y}_{[i]}^0$ due to (4.27) as

$$\begin{aligned}
\mathbf{K}_i(t_0) &= \mathbf{U}_i^0\, \mathbf{S}_i^0 = \mathbf{Y}_{[i]}^0\, \mathbf{V}_i^0, \\
\mathbf{S}_i(t_0) &= \mathbf{L}_i^{0,\top}\, \mathbf{V}_i^0 = \mathbf{Mat}_i\Big(A(t_0) \bigtimes_{k=1}^{i} \mathbf{U}_k^{1,\top}\Big)\, \mathbf{V}_i^0 = \mathbf{U}_i^{1,\top}\, \mathbf{Mat}_i\Big(A(t_0) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big)\, \mathbf{V}_i^0 \\
&= \mathbf{U}_i^{1,\top}\, \mathbf{Y}_{[i]}^0\, \mathbf{V}_i^0, \\
\mathbf{L}_i(t_0)^\top &= \mathbf{Mat}_i\Big(A(t_0) \bigtimes_{k=1}^{i} \mathbf{U}_k^{1,\top}\Big) = \mathbf{U}_i^{1,\top}\, \mathbf{Mat}_i\Big(A(t_0) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big) = \mathbf{U}_i^{1,\top}\, \mathbf{Y}_{[i]}^0\,.
\end{aligned}$$

Now, the substep of Algorithm 6 in the i-mode unfolding solves exactly the differential equations

$$\begin{aligned}
\dot{\mathbf{K}}_i(t) &= \mathbf{Mat}_i\Big(\dot{A}(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big)\, \mathbf{V}_i^0, \qquad \mathbf{K}_i(t_0) = \mathbf{Y}_{[i]}^0\, \mathbf{V}_i^0 \\
\dot{\mathbf{S}}_i(t) &= -\,\mathbf{U}_i^{1,\top}\, \mathbf{Mat}_i\Big(\dot{A}(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big)\, \mathbf{V}_i^0, \qquad \mathbf{S}_i(t_0) = \mathbf{U}_i^{1,\top}\, \mathbf{Y}_{[i]}^0\, \mathbf{V}_i^0,
\end{aligned}$$

and approximately the differential equation

$$\dot{\mathbf{L}}_i(t)^\top = \mathbf{U}_i^{1,\top}\, \mathbf{Mat}_i\Big(\dot{A}(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big), \qquad \mathbf{L}_i(t_0)^\top = \mathbf{U}_i^{1,\top}\, \mathbf{Y}_{[i]}^0\,.$$

Since $\mathbf{Y}^1_{[i+1]}$ is the exact solution for the $(i+1)$-mode setting, we conclude by induction hypothesis

$$\begin{aligned}\mathbf{L}_i^{1,\top} &= \mathbf{Mat}_i\big(\mathrm{Ten}_{i+1}(\mathbf{Y}^1_{[i+1]})\big) = \mathbf{Mat}_i\Big(A(t_1) \bigtimes_{k=1}^{i} \mathbf{U}_k^{1,\top}\Big) \\ &= \mathbf{U}_i^{1,\top}\, \mathbf{Mat}_i\Big(A(t_1) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big) = \mathbf{L}_i(t_1)^\top.\end{aligned}$$

Hence also the differential equation in the third substep of the i-mode unfolded subproblem is solved exactly. It follows by the exactness result for the matrix projector-splitting integrator that the nested Tucker integrator solves (4.25) with initial value (4.26) exactly, so that (4.29) is satisfied. Now, within the integrator, we transform the considered differential equations from mode $(i-1)$ to mode i by setting

$$\mathbf{Y}_{[i]} = \mathbf{Mat}_i\big(\mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^\top)\big),$$

which is equivalent to

$$\mathrm{Ten}_i(\mathbf{Y}_{[i]}) = \mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^\top). \tag{4.30}$$

Hence, with (4.29) and (4.30), we finally obtain

$$\begin{aligned}Y^1 &= \mathrm{Ten}_1(\mathbf{U}_1^1\, \mathbf{L}_1^{1,\top}) = \mathrm{Ten}_1(\mathbf{L}_1^{1,\top}) \times_1 \mathbf{U}_1^1 = \mathrm{Ten}_2(\mathbf{Y}^1_{[2]}) \times_1 \mathbf{U}_1^1 \\ &= \mathrm{Ten}_2(\mathbf{U}_2^1\, \mathbf{L}_2^{1,\top}) \times_1 \mathbf{U}_1^1 = \mathrm{Ten}_2(\mathbf{L}_2^{1,\top}) \times_2 \mathbf{U}_2^1 \times_1 \mathbf{U}_1^1 = \mathrm{Ten}_3(\mathbf{Y}^1_{[3]}) \times_2 \mathbf{U}_2^1 \times_1 \mathbf{U}_1^1 \\ &\;\;\vdots \\ &= \mathrm{Ten}_i(\mathbf{U}_i^1\, \mathbf{L}_i^{1,\top}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 \\ &\;\;\vdots \\ &= \mathrm{Ten}_d(\mathbf{U}_d^1\, \mathbf{L}_d^{1,\top}) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^1 \\ &= \mathrm{Ten}_d(\mathbf{Y}^1_{[d]}) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^1 = \mathrm{Ten}_{d-1}(\mathbf{L}_{d-1}^{1,\top}) \bigtimes_{k=1}^{d-1} \mathbf{U}_k^1 = \mathrm{Ten}_{d-1}(\mathbf{U}_{d-1}^1\, \mathbf{L}_{d-1}^{1,\top}) \bigtimes_{k=1}^{d-2} \mathbf{U}_k^1 \\ &= \mathrm{Ten}_{d-1}(\mathbf{Y}^1_{[d-1]}) \bigtimes_{k=1}^{d-2} \mathbf{U}_k^1 \\ &\;\;\vdots \\ &= \mathrm{Ten}_i(\mathbf{Y}^1_{[i]}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 \\ &\;\;\vdots \\ &= \mathrm{Ten}_1(\mathbf{Y}^1_{[1]}) = \mathrm{Ten}_1\big(\mathbf{Mat}_1(A(t_1)\big) \\ &= A(t_1),\end{aligned}$$

which shows the exactness of the nested Tucker integrator. □

4.5 Error bounds for the nested Tucker integrator

We now show that the nested Tucker integrator is robust in the presence of small singular values, see [LVW18, Theorem 5.1]. Since the integrator is based on recursively applying the matrix projector-splitting integrator, the plan is to analyze these recursive steps from the matrix perspective so that we can apply Theorem 2.4. To this end, we first need to generalize the assumptions of Theorem 2.4.

Let $A(t)$ be the solution of (4.1), viz.,

$$\dot{A}(t) = F(t, A(t)), \qquad A(t_0) = A^0$$

on $[t_0, T]$. Recall that $\mathcal{M}$ is the manifold of tensors of multilinear rank $(r_1, \ldots, r_d)$. Let

$$\mathcal{M}_i = \{Y \in \mathbb{R}^{n_1 \times \cdots \times n_d} \colon \operatorname{rank}(\mathbf{Mat}_i(Y)) = r_i\},$$

so that $\mathcal{M} = \mathcal{M}_1 \cap \cdots \cap \mathcal{M}_d$.

Assumption 4.2. *We assume that for each $i = 1, \ldots, d$, the i-mode unfolding of (4.1) satisfies the following conditions:*

(1) $F(t, Y)$ is Lipschitz continuous:

$$\|F(t, Y) - F(t, \widetilde{Y})\| \leq L\|Y - \widetilde{Y}\|, \qquad \text{for all} \quad Y, \widetilde{Y} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$$

(2) $F(t, Y)$ is bounded:

$$\|F(t, Y)\| \leq B \qquad \text{for all} \quad Y \in \mathbb{R}^{n_1 \times \cdots \times n_d} \tag{4.31}$$

(3) $F(t, Y)$ can be decomposed into a tangential part and a small perturbation:

$$\begin{aligned} &F(t, Y) = M_i(t, Y) + R_i(t, Y), \\ &M_i(t, Y) \in \mathcal{T}_Y \mathcal{M}_i, \quad \|R_i(t, Y)\| \leq \varepsilon, \end{aligned} \tag{4.32}$$

for all $Y \in \mathcal{M}_i$ in a neighborhood of $A(t)$ and for all $t \in [t_0, T]$,

(4) the initial value $A(t_0)$ for (4.1) has multilinear rank $(r_1, \ldots, r_d)$.

The condition (4.32) is formulated in terms of $\mathcal{M}_i$ that are essentially fixed matrix manifolds. Since we are solving (4.2) on a fixed rank Tucker manifold $\mathcal{M}$, it seems more natural to impose that $F(t, Y)$ is close to the tangent space of $\mathcal{M}$ that is,

$$\|F(t, Y) - \mathrm{P}(Y)F(t, Y)\| \leq \varepsilon. \tag{4.33}$$

However, since $\mathcal{M} = \mathcal{M}_1 \cap \cdots \cap \mathcal{M}_d$, by definition of a tangent space we get $T_Y\mathcal{M} \subseteq T_Y\mathcal{M}_1 \cap \cdots \cap T_Y\mathcal{M}_d$ for $Y \in \mathcal{M}$. Hence, $\mathrm{P}(Y)F(t, Y) \in T_Y\mathcal{M}_i$ for all $i = 1, \ldots, d$ and so (4.33) actually implies (4.32) for all $Y \in \mathcal{M}$.

Theorem 4.3. *Under assumption 4.2, the error of the nested Tucker integrator after n steps with step size $h > 0$ satisfies for all $t_n = t_0 + nh \leq T$:*

$$\|Y_n - A(t_n)\| \leq c_1 h + c_2 \varepsilon,$$

where the constants c_1 and c_2 only depend on L, B, T and the dimension d. In particular, the constants are independent of singular values of matricizations of the exact or approximate solution tensor.

Proof. Recall from (4.18) and (4.20) for the derivation of the nested Tucker integrator that Algorithm 6 solves approximately the following subproblems for each mode i on $\mathbb{R}^{r_1 \times \cdots \times r_{i-1} \times n_i \times \cdots \times n_d}$:

$$\dot{Y}_i(t) = F\Big(t, Y_i(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\Big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}, \qquad Y_i(t_0) = \mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,\top})$$

with $\mathbf{L}_{i-1}^{0,\top} = \widetilde{\mathbf{S}}_{i-1}^0 \, \mathbf{V}_{i-1}^{0,\top}$. Introducing

$$Z_i(t) = Y_i(t) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 \ \in \mathcal{M}_1 \cap \cdots \cap \mathcal{M}_{i-1} \subset \mathbb{R}^{n_1 \times \cdots \times n_d},$$

we obtain the equivalent initial value problem

$$\dot{Z}_i(t) = F\big(t, Z_i(t)\big) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \, \mathbf{U}_k^{1,\top}), \qquad Z_i(t_0) = \mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,\top}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 \tag{4.34}$$

on $\mathbb{R}^{n_1 \times \cdots \times n_d}$. We note that since $\mathbf{L}_{i-1}^{0,\top}$ has full rank, we have $Z_i(t_0) \in \mathcal{M}_i$. The nested Tucker integrator solves the i-mode unfolded subproblems

$$\begin{aligned} \dot{\mathbf{Z}}_{[i]}(t) &= \mathbf{Mat}_i\Big(F(t, Z_i(t)) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \, \mathbf{U}_k^{1,\top})\Big), \\ \mathbf{Z}_{[i]}(t_0) &= \mathbf{Mat}_i\Big(\mathrm{Ten}_{i-1}(\mathbf{L}_{i-1}^{0,\top}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big), \end{aligned} \tag{4.35}$$

by applying the matrix projector-splitting integrator onto

$$\begin{aligned} \dot{\mathbf{K}}_i(t) &= \mathbf{Mat}_i\Big(F(t, Z_i(t)) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \, \mathbf{U}_k^{1,\top})\Big) \, \mathbf{V}_i^0, \qquad \mathbf{K}_i^0 = \mathbf{Y}_i^0 \, \mathbf{V}_i^0, \\ \dot{\mathbf{S}}_i(t) &= -\,\mathbf{U}_i^{1,\top} \, \mathbf{Mat}_i\Big(F(t, Z_i(t)) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \, \mathbf{U}_k^{1,\top})\Big) \, \mathbf{V}_i^0, \qquad \mathbf{S}_i^0 = \mathbf{U}_i^{0,\top} \, \mathbf{Y}_i^0 \, \mathbf{V}_i^0, \\ \dot{\mathbf{L}}_i(t)^\top &= \mathbf{U}_i^{1,\top} \, \mathbf{Mat}_i\Big(F(t, Z_i(t)) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \, \mathbf{U}_k^{1,\top})\Big), \qquad \mathbf{L}_i^{0,\top} = \mathbf{U}_i^{0,\top} \, \mathbf{Y}_i^0, \end{aligned} \tag{4.36}$$

where the **K**- and the **S**-step are solved exactly, but the **L**-step is solved inexactly for all $i = 1, \ldots, d-1$. Instead, for $i = d$, all three substeps (4.36) are solved exactly. Hence, this results in the approximation $Z_i^1 \in \mathcal{M}_i$ to $Z_i(t_1)$.

The solution to the **KSL**-ODE system (4.36) is determined by the integration method for matrices, where a rigorous error analysis already exists, see Section 2.3. Now, in order to make use of it, we have to make sure that (4.34) and the above conditions *(1)-(4)* follow the Assumption 2.3 for the proof of the matrix case.

(1) Lipschitz continuity of the right-hand side of (4.34):
Let $Z_i(t), \widetilde{Z}_i(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, then, since $\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}$ is an orthogonal projection, we find

$$\begin{aligned}
&\Big\| F(t, Z_i(t)) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) - F(t, \widetilde{Z}_i(t)) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \Big\| \\
&= \Big\| \mathbf{Mat}_i\big(F(t, Z_i(t)) - F(t, \widetilde{Z}_i(t))\big) \Big(\big(\bigotimes_{k=1}^{i-1} \mathbf{U}_k^1 \mathbf{U}_k^{1,\top} \big) \otimes \big(\bigotimes_{k=i}^{d} \mathbf{I}_{r_k} \big) \Big) \Big\| \\
&= \| F(t, Z_i(t)) - F(t, \widetilde{Z}_i(t)) \| \\
&\leq L \| Z_i(t) - \widetilde{Z}_i(t) \|,
\end{aligned}$$

since by assumption $F(t, Y)$ is Lipschitz continuous for all $Y \in \mathbb{R}^{n_1 \times \cdots \times n_d}$.

(2) Boundedness of the right-hand side of (4.34):
Again, since $\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}$ is an orthogonal projection for each mode $k = 1, \ldots, i-1$ and for all $i = 1, \ldots, d$, the boundedness follows trivially from the assumption (4.31): for $Z_i(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, we have

$$\begin{aligned}
\Big\| F(t, Z_i(t)) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \Big\| &= \Big\| \mathbf{Mat}_i\big(F(t, Z_i(t))\big) \Big(\big(\bigotimes_{k=1}^{i-1} \mathbf{U}_k^1 \mathbf{U}_k^{1,\top} \big) \otimes \big(\bigotimes_{k=i}^{d} \mathbf{I}_{r_k} \big) \Big) \Big\| \\
&\leq \| F(t, Z_i(t)) \| \\
&\leq B.
\end{aligned}$$

(3) For each mode i, the right-hand side of (4.34) lies in the tangent space up to a small remainder:
By assumption (4.32), the i-mode unfolding of the right-hand side of (4.34) can be decomposed, for any $Y \in \mathcal{M}_i$, as

$$\begin{aligned}
\mathbf{Mat}_i\Big(F(t, Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \Big) &= \mathbf{Mat}_i\Big(M_i(t, Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \Big) \\
&\quad + \mathbf{Mat}_i\Big(R_i(t, Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \Big),
\end{aligned}$$

where by assumption (4.32) $M_i(t, Y) \in \mathcal{T}_Y \mathcal{M}_i$ and $\| R_i(t, Y) \| \leq \varepsilon$, for all $i = 1, \ldots, d$.

In order to be able to follow the pattern of the proof of the error results from the matrix case shown in Section 2.3, we have to assure that $\mathbf{Mat}_i\big(M_i(t, Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \big)$ is in the tangent space of the matricized manifold $\mathcal{M}$. First, we show that

$$M_i(t, Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \in \mathcal{T}_Y \mathcal{M}_i,$$

afterwards we analyze the corresponding matrix setting. Now, suppose $M_i(t, Y) \in \mathcal{T}_Y \mathcal{M}_i$ and

$$Y = Y \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}), \tag{4.37}$$

i.e. the orthogonal projections $\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}$ consisting of the already updated basis matrices for the modes $1, \dots, i-1$ are not influential regarding the approximation tensor. This holds true for $Y = Z_i(t)$ in (4.34). Considering the SVD of $\mathbf{Mat}_i(Y)$ and due to (4.37) we obtain

$$\begin{aligned}\mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^\top = \mathbf{Mat}_i(Y) = \mathbf{Mat}_i\Big(Y \overset{i-1}{\underset{k=1}{\times}} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top})\Big) = \mathbf{Mat}_i(Y) \bigotimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \\ = \mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^\top \bigotimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}),\end{aligned}$$

and so

$$\mathbf{V}_i^\top = \mathbf{V}_i^\top \bigotimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}). \tag{4.38}$$

Now, similarly as in Section 1.2, we know that $M_i(t, Y) \in \mathcal{T}_Y \mathcal{M}_i$ implies

$$\mathbf{Mat}_i(M_i(t,Y)) = \delta \mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^\top + \mathbf{U}_i \delta \mathbf{S}_i \mathbf{V}_i^\top + \mathbf{U}_i \mathbf{S}_i \delta \mathbf{V}_i^\top,$$

and so with (4.38) it follows that

$$\begin{aligned}\mathbf{Mat}_i\Big(M_i(t,Y) \overset{i-1}{\underset{k=1}{\times}} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top})\Big) &= \mathbf{Mat}_i\big(M_i(t,Y)\big) \bigotimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \\ &= \delta \mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^\top \bigotimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \\ &\quad + \mathbf{U}_i \delta \mathbf{S}_i \mathbf{V}_i^\top \bigotimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \\ &\quad + \mathbf{U}_i \mathbf{S}_i \delta \mathbf{V}_i^\top \bigotimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \\ &= \delta \mathbf{U}_i \mathbf{S}_i \mathbf{V}_i^\top + \mathbf{U}_i \delta \mathbf{S}_i \mathbf{V}_i^\top + \mathbf{U}_i \mathbf{S}_i \delta \mathbf{V}_i^\top,\end{aligned}$$

with some modified $\delta \mathbf{V}_i$. Therefore, $\mathbf{Mat}_i\Big(M_i(t,Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top})\Big)$ is of the same form as $\mathbf{Mat}_i(M_i(t,Y))$, which implies $M_i(t,Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}) \in \mathcal{T}_Y \mathcal{M}_i$. Further, the manifold of matrices of rank r_i of dimension $n_i \times n_1 \cdots n_{i-1} \cdot n_{i+1} \cdots n_d$ is given as

$$\mathbf{Mat}_i(\mathcal{M}_i) = \{\mathbf{Mat}_i(Y) \mid Y \in \mathcal{M}_i\}.$$

Moreover, $Y \in \mathcal{M}_i$ and denote $\mathbf{Y} := \mathbf{Mat}_i(Y) \in \mathbf{Mat}_i(\mathcal{M}_i)$. Then we have

$$\mathcal{T}_{\mathbf{Y}} \mathbf{Mat}_i(\mathcal{M}_i) = \mathbf{Mat}_i(\mathcal{T}_Y \mathcal{M}_i)$$

and so we conclude that

$$\mathbf{Mat}_i\Big(M_i(t,Y) \overset{i-1}{\underset{k=1}{\times}} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top})\Big) \in \mathcal{T}_{\mathbf{Y}} \mathbf{Mat}_i(\mathcal{M}_i).$$

Thanks to assumption (4.32) and since $\mathbf{U}_k^1 \mathbf{U}_k^{1,\top}$ are orthogonal projections for each $k = 1, \ldots, i-1$, the perturbation term is controlled by

$$\begin{aligned}\Big\| \mathbf{Mat}_i\Big(R_i(t,Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top})\Big)\Big\| &= \Big\| R_i(t,Y) \bigtimes_{k=1}^{i-1} (\mathbf{U}_k^1 \mathbf{U}_k^{1,\top})\Big\| \\ &\leq \|R_i(t,Y)\| \\ &\leq \varepsilon.\end{aligned}$$

(4) Verifying the rank of the initial value:
We know that $Z_i(t_0) \in \mathcal{M}_i$. This implies $\operatorname{rank} \mathbf{Mat}_i(Z_i(t_0)) = r_i$, for all $i = 1, \ldots, d$.

To show the error bound, let us first consider the case of mode d. Here, the nested Tucker integrator applies the matrix projector-splitting integrator, which in turn solves all three steps exactly. After having verified the assumptions, the error analysis of the matrix case given in Section 2.3 applied to the problem (4.35) for $i = d$, which is equivalent to the subproblems within the nested Tucker integrator, can be applied and yields the error bound

$$\|Z_d^1 - Z_d(t_1)\| = \mathcal{O}(h(\varepsilon + h)),$$

where the constants symbolized by the $\mathcal{O}$ notation depend only on L, B and d.

In the modal step $d-1$, the integrator applies the matrix integrator with exact solution of the first two, but, due to the low-rank approximation, inexact solution in the third step of the problems (4.36) to be solved. The error of this inexact solution is given by $\mathcal{O}(h\eta)$, where $\eta = \varepsilon + h$, because by construction of the integration scheme, the approximate solution of the third step in the $(d-1)$-modal step is given by the solution of the full mode-d step. From Section 2.4.2, we know the local error of the matrix projector-splitting integrator with inexact solution in the substeps and so the error in mode $(d-1)$ is given by

$$\|Z_{d-1}^1 - Z_{d-1}(t_1)\| = \mathcal{O}(h(\varepsilon + h + \eta)) = \mathcal{O}(h(\varepsilon + h)). \tag{4.39}$$

Using these error bounds for d and $(d-1)$, we show by induction for $d-2, \ldots, 1$ the local error bound

$$\|Z_i^1 - Z_i(t_1)\| = \mathcal{O}(h(\varepsilon + h)). \tag{4.40}$$

Suppose, this has been shown for modes $d-2, \ldots, i+1$. Then, we apply the matrix projector-splitting algorithm to the i-th unfolding with an inexact solution of the third substep. The error of this inexact solution is given by (4.40) for $(i+1)$. We conclude that the error bound for the i-th step is of the form (4.40). With the induction hypothesis that (4.40) holds for $i+1, \ldots, d$, we conclude from the error bound (4.39) that (4.40) also holds for mode i.

Finally, for $i = 1$, we have $Z_1(t) = Y_1(t)$, such that the local error of the nested Tucker integrator is given by

$$\|Y^1 - Y_1(t_1)\| = \mathcal{O}(h(\varepsilon + h)).$$

Following the pattern of propagating the error until final time step $t_n = nh$ and adding the transported errors up, see [HNW93, Lady Windermere's fan], we obtain the stated error bound. □

4.6 A projector-splitting integrator for Tucker tensors

In this section, we discuss an integration method proposed by Lubich in [Lub15] and compare it with the nested Tucker integrator. To give an insight behind the scene, the evolutionary history of the nested Tucker integrator started with the difficulty of proving an exactness result for the integrator presented in [Lub15], such as given for the matrix case and in Section 4.4. Now, by having proven the exactness property of the nested Tucker integrator in Section 4.4 and showing mathematical equivalence of those integrators in Section 4.6.5, we can conclude exactness also for the integration method in [Lub15]. However, after having thought about the integrator [Lub15] again, but with the focus on the projections that appear within the substeps, we found a direct proof for its exactness, which we present in Section 4.6.3. This proof is not published nor submitted elsewhere.

We first recall the integration scheme of [Lub15] in Section 4.6.1, give the new proof about the exactness afterwards in Section 4.6.3 and discuss the differences between the two integrators. Finally, we prove the equivalence of the two integration methods for Tucker tensors in Section 4.6.5.

4.6.1 Deriving the integration method

Here, we recall the integration method proposed in [Lub15]. We aim to compute a low-rank approximation to the tensor differential equation

$$\dot{A}(t) = F(t, A(t)), \qquad A(t_0) = A^0,$$

where $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, in the Tucker tensor format $Y(t) = C(t) \bigtimes_{i=1}^{d} \mathbf{U}_i(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ with $\operatorname{rank} Y = (r_1, \ldots, r_d)$. As in the derivation of the nested Tucker integrator, we first transform the tensor differential equation by matricizing in the first mode, which yields a matrix differential equation

$$\mathbf{Mat}_1\big(\dot{A}(t)\big) = \mathbf{Mat}_1\big(F(t, A(t))\big), \qquad \mathbf{Mat}_1(A(t_0)) = \mathbf{Mat}_1(A^0). \tag{4.41}$$

Suppose that $\mathbf{Mat}_1(Y^0) \in \mathcal{M}$ is a an approximation matrix to the initial value $\mathbf{Mat}_1(A^0)$, which is given by

$$\mathbf{Mat}_1(Y^0) = \mathbf{U}_1^0 \, \mathbf{Mat}_1(C^0) \bigotimes_{k=2}^{d} \mathbf{U}_k^{0,\top} = \mathbf{U}_1^0 \, \mathbf{S}_1^0 \, \mathbf{Q}_1^{0,\top} \bigotimes_{k=2}^{d} \mathbf{U}_k^{0,\top}, \tag{4.42}$$

where with

$$\mathbf{V}_1^{0,\top} = \mathbf{Q}_1^{0,\top} \bigotimes_{k=2}^{d} \mathbf{U}_k^{0,\top},$$

we obtain the necessary SVD-like decomposition

$$\mathbf{Mat}_1(Y^0) = \mathbf{U}_1^0 \, \mathbf{S}_1^0 \, \mathbf{V}_1^{0,\top}.$$

After having transformed the tensor into a matrix setting, the integrator applies the first two steps of the matrix projector-splitting integrator introduced in Section 2.1 for each modal step $i = 1, \ldots, d$ by solving

1. **K-step**: Update $\mathbf{U}_i^0 \to \mathbf{U}_i^1$, $\mathbf{S}_i^0 \to \widehat{\mathbf{S}}_i^1$ by solving

$$\begin{aligned}\dot{\mathbf{K}}_i(t) &= \mathbf{Mat}_i\big(F(t, \mathrm{Ten}_i(\mathbf{K}_i(t)\,\mathbf{V}_i^{0,\top}))\big)\,\mathbf{V}_i^0,\\ \mathbf{K}_i(t_0) &= \mathbf{U}_i^0\,\mathbf{S}_i^0\end{aligned}$$

2. **S-step**: Update $\widehat{\mathbf{S}}_i^1 \to \widetilde{\mathbf{S}}_i^0$ by solving

$$\begin{aligned}\dot{\mathbf{S}}_i(t) &= -\,\mathbf{U}_i^{1,\top}\,\mathbf{Mat}_i\big(F(t, \mathrm{Ten}_i(\mathbf{U}_i^1\,\mathbf{S}_i(t)\,\mathbf{V}_i^{0,\top}))\big)\,\mathbf{V}_i^0,\\ \mathbf{S}_i(t_0) &= \widehat{\mathbf{S}}_i^1.\end{aligned}$$

Hence, in order to update the first basis matrix, the integrator applies the above **KS**-scheme onto (4.41) with initial value (4.42) for $i = 1$. We point out that contrary to the Tucker integrator, this integrator does not solve the **L**-step within the matrix projector-splitting integrator (neither directly nor approximately). This arises from the idea that the basis matrix that needs to be updated is taken "in front" by matricizing the current approximation tensor Y, see (4.42), then the first two steps of the matrix integrator are applied and afterwards, the resulting updated basis matrix is shifted "in the back" again by retensorizing, see (4.43). So after each **KS**-step, an intermediate tensor $\overline{Y}_i$, for $i = 2, \ldots, d$ of size $n_1 \times \cdots \times n_d$ is constructed.

Now, updating $\mathbf{K}_1$ and $\mathbf{S}_1$ results in an intermediate approximation tensor

$$\overline{Y}_2(t) := C_2(t) \times_1 \mathbf{U}_1^1 \bigtimes_{k=2}^{d} \mathbf{U}_k(t) \quad \in \mathbb{R}^{n_1 \times \cdots \times n_d}, \tag{4.43}$$

which satisfies the tensor differential equation

$$\dot{\overline{Y}}_2(t) = F(t, \overline{Y}_2(t)), \qquad \overline{Y}_2(t_0) = C_2^0 \times_1 \mathbf{U}_1^1 \bigtimes_{k=2}^{d} \mathbf{U}_k^0,$$

where $C_2^0 = \mathrm{Ten}_1(\widetilde{\mathbf{S}}_1^0\,\mathbf{Q}_1^{0,\top})$. We continue with updating the basis matrix in the second mode first by preparing the factors through a QR decomposition

$$\mathbf{Mat}_2(C_2^0)^\top = \mathbf{Q}_2^0\,\mathbf{S}_2^{0,\top}$$

and setting

$$\mathbf{V}_2^{0,\top} = \mathbf{Q}_2^{0,\top}\Big(\mathbf{U}_1^{1,\top}\Big(\bigotimes_{k=3}^{d} \mathbf{U}_k^{0,\top}\Big)\Big).$$

For simplicity and analogy of notation we denote $\overline{\mathbf{Y}}_{[2]} := \mathbf{Mat}_2(\overline{Y}_2)$, and obtain the matrix differential equation

$$\begin{aligned}\dot{\overline{\mathbf{Y}}}_{[2]}(t) = \mathbf{Mat}_2\Big(F\big(t, \mathrm{Ten}_2(\overline{\mathbf{Y}}_{[2]}(t))\big)\Big), \qquad \overline{\mathbf{Y}}_{[2]}(t_0) &= \mathbf{U}_2^0\,\mathbf{Mat}_2(C_2^0)\Big(\mathbf{U}_1^{1,\top}\Big(\bigotimes_{k=3}^{d} \mathbf{U}_k^{0,\top}\Big)\Big)\\ &= \mathbf{U}_2^0\,\mathbf{S}_2^0\,\mathbf{V}_2^{0,\top},\end{aligned}$$

whose initial value again is of the required SVD-like factorization. Now that we have set up the matrix setting again, we solve the **K**- and the **S**-step from above for $i = 2$. This results in another approximation tensor

$$\overline{Y}_3(t) := C_3(t) \bigtimes_{l=1}^{2} \mathbf{U}_l^1 \bigtimes_{k=3}^{d} \mathbf{U}_i(t),$$

which solves

$$\dot{\overline{Y}}_3(t) = F(t, \overline{Y}_3(t)), \qquad \overline{Y}_3(t_0) = C_3^0 \bigtimes_{l=1}^{2} \mathbf{U}_l^1 \bigtimes_{k=3}^{d} \mathbf{U}_k^0,$$

where $C_3^0 = \mathrm{Ten}_2(\widetilde{\mathbf{S}}_2^0 \, \mathbf{Q}_2^{0,\top})$.

Proceeding analogously for $i = 3, \ldots, d$ yields updates $\mathbf{U}_i^0 \to \mathbf{U}_i^1$ as well as $\widehat{\mathbf{S}}_i^1 \to \widetilde{\mathbf{S}}_i^0$ and finally the intermediate approximation tensor $\overline{Y}_{d+1}(t) := C_{d+1}(t) \bigtimes_{l=1}^{d} \mathbf{U}_l^1$ with updated basis matrices but not yet updated core. So we are left with approximating the core tensor. To this end, we actually solve the tensor differential equation

$$\dot{C}(t) = F\Big(t, C(t) \bigtimes_{l=1}^{d} \mathbf{U}_l^1\Big) \bigtimes_{l=1}^{d} \mathbf{U}_l^{1,\top}, \qquad C(t_0) = C_{d+1}^0,$$

where as before $C_{d+1}^0 = \mathrm{Ten}_d(\widetilde{\mathbf{S}}_d^0 \, \mathbf{Q}_d^{0,\top})$.

Finally, this integration scheme results in the approximation tensor

$$C^1 \bigtimes_{l=1}^{d} \mathbf{U}_l^1 = Y^1 \approx Y(t_1) \approx A(t_1).$$

Algorithm 7 demonstrates how this integrator can be implemented.

Algorithm 7: One time step of the Tucker integrator of [Lub15]

Data: Tucker tensor $Y^0 = C^0 \bigtimes_{i=1}^{d} \mathbf{U}_i^0$, $F(t,Y)$, t_0, t_1
Result: Tucker tensor $Y^1 = C^1 \bigtimes_{i=1}^{d} \mathbf{U}_i^1$

begin
  for $i = 1$ **to** d **do**
    compute QR factorization $\mathbf{Mat}_i(C^0)^\top = \mathbf{Q}_i^0 \, \mathbf{S}_i^{0,\top}$
    set $\mathbf{V}_i^{0,\top} = \mathbf{Mat}_i\Big(\mathrm{Ten}_i(\mathbf{Q}_i^{0,\top}) \bigtimes_{l=1}^{i-1} \mathbf{U}_l^1 \bigtimes_{k=i+1}^{d} \mathbf{U}_k^0\Big)$
    set $\mathbf{K}_i^0 = \mathbf{U}_i^0 \, \mathbf{S}_i^0$
    set $\mathbf{Y}_{[i]}^+(t) = \mathbf{K}_i(t) \, \mathbf{V}_i^{0,\top}$
    solve $\dot{\mathbf{K}}_i(t) = \mathbf{Mat}_i\big(F(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^+))\big) \, \mathbf{V}_i^0$,
     with initial value $\mathbf{K}_i(t_0) = \mathbf{K}_i^0$ and return $\mathbf{K}_i^1 = \mathbf{K}_i(t_1)$
    compute QR factorization $\mathbf{K}_i^1 = \mathbf{U}_i^1 \, \widehat{\mathbf{S}}_i^1$
    set $\mathbf{Y}_{[i]}^-(t) = \mathbf{U}_i^1 \, \mathbf{S}_i(t) \, \mathbf{V}_i^{0,\top}$
    solve $\dot{\mathbf{S}}_i(t) = -\,\mathbf{U}_i^{1,\top} \, \mathbf{Mat}_i\big(F(t, \mathrm{Ten}_i(\mathbf{Y}_{[i]}^-))\big) \, \mathbf{V}_i^0$,
     with initial value $\mathbf{S}_i(t_0) = \widehat{\mathbf{S}}_i^1$ and return $\widetilde{\mathbf{S}}_i^0 = \mathbf{S}_i(t_1)$
    set $C^0 = \mathrm{Ten}_i(\widetilde{\mathbf{S}}_i^0 \, \mathbf{Q}_i^{0,\top})$
  solve $\dot{C}(t) = F\big(t, C(t) \bigtimes_{l=1}^{d} \mathbf{U}_l^1\big) \bigtimes_{l=1}^{d} \mathbf{U}_l^{1,\top}$,
   with initial value $C(t_0) = C^0$ and return $C^1 = C(t_1)$
  set $Y^1 = C^1 \bigtimes_{l=1}^{d} \mathbf{U}_l^1$

4.6.2 Interpretation as a projector-splitting integrator

In this section, we will analyze the orthogonal projection within the differential equation

$$\dot{Y}(t) = \mathrm{P}(Y(t)) F(t, Y(t)), \qquad Y(t_0) = Y^0,$$

where we lean on the result [Lub15, Sect. 6]. In [KL10, Lemma 3.1], O. Koch and Ch. Lubich state that the projector $\mathrm{P}(Y(t))$ can be written by a sum of subprojections. Assuming that the tensor $Y(t) \in \mathcal{M}$ is in Tucker format with $\mathbf{Mat}_i(Y) = \mathbf{U}_i \, \mathbf{S}_i \, \mathbf{V}_i^\top$ for all $i = 1, \ldots, d$, the explicit formula for the tangent space projection $\mathrm{P}(Y)$, leaving out the time dependence, is given as

$$\begin{aligned} \mathrm{P}(Y) = \sum_{i=1}^{d} \mathrm{Ten}_i\Big((\mathbf{I}_{n_i} - \mathbf{U}_i \, \mathbf{U}_i^\top) \, \mathbf{Mat}_i(F(t,Y)) \bigotimes_{k \neq i} \mathbf{U}_k \, \mathbf{Mat}_i(C_{i+1}^\dagger) \, \mathbf{Mat}_i(C_{i+1}) \bigotimes_{k \neq i} \mathbf{U}_k^\top \Big) \\ + \; F(t,Y) \bigtimes_{i=1}^{d} \mathbf{U}_i \, \mathbf{U}_i^\top, \end{aligned} \tag{4.44}$$

where

$$C_{i+1}^\dagger = \mathbf{Mat}_i(C_{i+1})^\top \big(\mathbf{Mat}_i(C_{i+1}) \, \mathbf{Mat}_i(C_{i+1})^\top \big)^{-1}$$

is the pseudo-inverse of the matricized core tensors C_{i+1} for all $i = 1, \dots, d$. From the description of the integrator in Section 4.6.1, we have $\mathbf{Mat}_i(C_{i+1}) = \widetilde{\mathbf{S}}_i \, \mathbf{Q}_i^\top$, where $\mathbf{Q}_i$ is an orthonormal matrix for each mode $i = 1, \dots, d$. Therefore, we find

$$\begin{aligned}
\mathbf{Mat}_i(C_{i+1}^\dagger) \, \mathbf{Mat}_i(C_{i+1}) &= \mathbf{Q}_i \, \widetilde{\mathbf{S}}_i^\top \big(\widetilde{\mathbf{S}}_i \, \mathbf{Q}_i^\top (\widetilde{\mathbf{S}}_i \, \mathbf{Q}_i^\top)^\top \big)^{-1} \widetilde{\mathbf{S}}_i \, \mathbf{Q}_i^\top \\
&= \mathbf{Q}_i \, \widetilde{\mathbf{S}}_i^\top \big(\widetilde{\mathbf{S}}_i \widetilde{\mathbf{S}}_i^\top \big)^{-1} \widetilde{\mathbf{S}}_i \, \mathbf{Q}_i^\top \\
&= \mathbf{Q}_i \, \mathbf{Q}_i^\top \, .
\end{aligned}$$

Inserting this into (4.44), we can simplify this equation to

$$\begin{aligned}
\mathrm{P}(Y) &= \sum_{i=1}^{d} \mathrm{Ten}_i \Big((\mathbf{I}_{n_i} - \mathbf{U}_i \, \mathbf{U}_i^\top) \, \mathbf{Mat}_i(F(t,Y)) \bigotimes_{k \neq i} \mathbf{U}_k \, \mathbf{Mat}_i(C_{i+1}^\dagger) \, \mathbf{Mat}_i(C_{i+1}) \bigotimes_{k \neq i} \mathbf{U}_k^\top \Big) \\
&\quad + \; F(t,Y) \bigtimes_{i=1}^{d} \mathbf{U}_i \, \mathbf{U}_i^\top \\
&= \sum_{i=1}^{d} \mathrm{Ten}_i \Big((\mathbf{I}_{n_i} - \mathbf{U}_i \, \mathbf{U}_i^\top) \, \mathbf{Mat}_i(F(t,Y)) \bigotimes_{k \neq i} \mathbf{U}_k \, \mathbf{Q}_i \, \mathbf{Q}_i^\top \bigotimes_{k \neq i} \mathbf{U}_k^\top \Big) \\
&\quad + \; F(t,Y) \bigtimes_{i=1}^{d} \mathbf{U}_i \, \mathbf{U}_i^\top \\
&= \sum_{i=1}^{d} \mathrm{Ten}_i \Big((\mathbf{I}_{n_i} - \mathbf{U}_i \, \mathbf{U}_i^\top) \, \mathbf{Mat}_i(F(t,Y)) \, \mathbf{V}_i \, \mathbf{V}_i^\top \Big) \\
&\quad + \; F(t,Y) \bigtimes_{i=1}^{d} \mathbf{U}_i \, \mathbf{U}_i^\top ,
\end{aligned}$$

where we have used that

$$\mathbf{V}_i = \bigotimes_{k \neq i} \mathbf{U}_k \, \mathbf{Q}_i ,$$

which comes from the matricization of the approximation tensor after having performed a QR decomposition of the matricized core, see (4.10). By denoting the appearing orthogonal projections as

$$\begin{aligned}
\mathrm{P}_i^+(Y) F(t,Y) &= \mathrm{Ten}_i \big(\mathbf{Mat}_i(F(t,Y)) \, \mathbf{V}_i \, \mathbf{V}_i^\top \big) , \\
\mathrm{P}_i^-(Y) F(t,Y) &= \mathrm{Ten}_i \big(\mathbf{U}_i \, \mathbf{U}_i^\top \, \mathbf{Mat}_i(F(t,Y)) \, \mathbf{V}_i \, \mathbf{V}_i^\top \big) , \\
\mathrm{P}_C(Y) F(t,Y) &= F(t,Y) \bigtimes_{i=1}^{d} \mathbf{U}_i \, \mathbf{U}_i^\top ,
\end{aligned}$$

we conclude that the orthogonal projection $\mathrm{P}(Y)$ onto $\mathcal{T}_Y \mathcal{M}$ is given as

$$\mathrm{P}(Y) = \sum_{i=1}^{d} \big(\mathrm{P}_i^+(Y) - \mathrm{P}_i^-(Y) \big) + \mathrm{P}_C(Y) . \tag{4.45}$$

Therefore, we interpret the Tucker integrator presented in Section 4.6 as a projector-splitting integrator. We follow the Lie–Trotter projector-splitting integrator and solve the

following differential equations consecutively for $i = 1, \dots, d$:

$$\begin{aligned} \dot{Y}_i^+(t) &= \mathrm{P}_i^+(Y_i^+(t))F(t, Y_i^+(t)), & Y_i^+(t_0) &= Y_{i-1}^-(t_1), \\ \dot{Y}_i^-(t) &= -\mathrm{P}_i^-(Y_i^-(t))F(t, Y_i^-(t)), & Y_i^-(t_0) &= Y_i^+(t_1), \\ \dot{Y}_C(t) &= \mathrm{P}_C(Y_C(t))F(t, Y_C(t)), & Y_C(t_0) &= Y_d^-(t_1), \end{aligned} \tag{4.46}$$

where for notational convenience $Y_0^-(t_1) = Y(t_0)$. Similarly as for the matrix case discussed in Section 2.1.1, we can determine closed-form solutions in the explicit case, i.e., when choosing $F(t, Y_i^\pm(t))$ and $F(t, Y_C(t))$ to be $\dot{A}(t)$, which will be given in the subsequent lemma. In this situation, the method just uses the increment ΔA instead of $\dot{A}(t)$.

For ease of presentation, we introduce the notation

$$\begin{aligned} \mathrm{P}_{\mathbf{V}}\Delta A &= \mathrm{Ten}_i\big(\mathbf{Mat}_i(\Delta A)\,\mathbf{V}_i(t_0)\,\mathbf{V}_i(t_0)^\top\big), \\ \mathrm{P}_{\mathbf{U}}\mathrm{P}_{\mathbf{V}}\Delta A &= \mathrm{Ten}_i\big(\mathbf{U}_i(t_1)\,\mathbf{U}_i(t_1)^\top\,\mathbf{Mat}_i(\Delta A)\,\mathbf{V}_i(t_0)\,\mathbf{V}_i(t_0)^\top\big), \\ \mathrm{P}_{\mathbf{W}}\Delta A &= \mathrm{Ten}_i\big(\mathbf{Mat}_i(\Delta A)\,\mathbf{W}_i(t_0)\,\mathbf{W}_i(t_0)^\top\big), \end{aligned}$$

where the factor matrices $\mathbf{U}_i, \mathbf{V}_i, \mathbf{W}_i$ come from the decomposition $\mathbf{Mat}_i(Y) = \mathbf{U}_i\,\mathbf{S}_i\,\mathbf{V}_i^\top = \mathbf{U}_i\,\mathbf{W}_i^\top$ for all $i = 1, \dots, d$.

Lemma 4.4. *Let $\Delta A = A(t_1) - A(t_0)$ and suppose $\mathbf{U}_i$ is orthogonal, for each $i = 1, \dots, d$. Then the subproblems* (4.46) *for the explicit case satisfy the following:*

$$\begin{aligned} &Y_i^+(t_0) = \mathrm{Ten}_i(\mathbf{U}_i^+(t_0)\,\mathbf{W}_i^+(t_0)^\top) \quad \textit{yields} \quad Y_i^+(t_1) = Y_i^+(t_0) + \mathrm{P}_i^+\Delta A, \\ &Y_i^-(t_0) = \mathrm{Ten}_i(\mathbf{U}_i^-(t_0)\,\mathbf{W}_i^-(t_0)^\top) \quad \textit{yields} \quad Y_i^-(t_1) = Y_i^-(t_0) - \mathrm{P}_i^-\Delta A, \\ &\qquad \textit{where } \mathrm{P}_i^+\Delta A = \mathrm{P}_{\mathbf{W}}\Delta A \quad \textit{and} \quad \mathrm{P}_i^-\Delta A = \mathrm{P}_{\mathbf{U}}\mathrm{P}_{\mathbf{W}}\Delta A. \end{aligned}$$

Proof. Before showing the first statement, we perform a QR decomposition $\mathbf{V}(t)\mathbf{S}(t)^\top = \mathbf{W}(t)$ and obtain for the initial value the SVD-like factorization $Y_i^\pm = \mathrm{Ten}_i(\mathbf{U}_i^\pm\,\mathbf{S}_i^\pm\,\mathbf{V}_i^{\pm,\top})$ with orthogonal matrices $\mathbf{U}_i^\pm$ and $\mathbf{V}_i^\pm$, respectively. For notational convenience, we will drop the indices and the superscripts of the factor matrices, but remember that they differ according to the substep. Since $Y_i^\pm(t) \in \mathcal{M}$ for all $i = 1, \dots, d$ and from the differential equation $\dot{Y}_i^+ = \mathrm{P}_i^+(Y_i^+)\dot{A}(t)$ we then find

$$\begin{aligned} \dot{Y}_i^+(t) &= \mathrm{Ten}_i(\dot{\mathbf{U}}\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\dot{\mathbf{S}}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\dot{\mathbf{V}}^\top), \\ \mathrm{P}_i^+(Y_i^+(t))\dot{A}(t) &= \mathrm{Ten}_i(\mathbf{Mat}_i(\dot{A}(t))\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top). \end{aligned}$$

Hence

$$(\dot{\mathbf{U}\mathbf{S}})\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\dot{\mathbf{V}}^\top = \mathbf{Mat}_i(\dot{A}(t))\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top,$$

which holds true for

$$(\dot{\mathbf{U}\mathbf{S}}) = \mathbf{Mat}_i(\dot{A}(t))\,\mathbf{V}(t_0) \qquad \text{and} \qquad \dot{\mathbf{V}} = \mathbf{0}.$$

Solving this differential equation gives after one time step $t_0 \to t_1$

$$\mathbf{U}(t_1)\mathbf{S}(t_1) = \mathbf{Mat}_i(\Delta A)\,\mathbf{V}(t_0)$$

and so

$$Y_i^+(t_1) = Y_i(t_0) + \mathrm{Ten}_i(\mathbf{Mat}_i(\Delta A)\,\mathbf{V}(t_0)\,\mathbf{V}^\top(t_0)) = Y_i(t_0) + \mathrm{P}_{\mathbf{V}}\Delta A.$$

Further, since $\mathrm{span}(\mathbf{V}(t)) = \mathrm{span}(\mathbf{W}(t))$ we conclude $\mathrm{P}_{\mathbf{V}} = \mathrm{P}_{\mathbf{W}} = \mathrm{P}_i^+$, which yields the stated result.
For the i^--substeps, we proceed in an analogous way. The differential equation for $Y_i^-(t)$ gives

$$\begin{aligned}\dot{Y}_i^-(t) &= \mathrm{Ten}_i\big(\dot{\mathbf{U}}\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\dot{\mathbf{S}}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\dot{\mathbf{V}}^\top\big),\\ \mathrm{P}_i^-(Y_i^-(t))\dot{A}(t) &= \mathrm{Ten}_i\big(\mathbf{U}(t_1)\,\mathbf{U}(t_1)^\top\,\mathbf{Mat}_i(\dot{A}(t))\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top\big)\end{aligned}$$

and hence

$$\dot{\mathbf{U}}\mathbf{S}\,\mathbf{V}^\top + \mathbf{U}\,\dot{\mathbf{S}}\,\mathbf{V}^\top + \mathbf{U}\,\mathbf{S}\dot{\mathbf{V}}^\top = \mathbf{U}(t_1)\,\mathbf{U}(t_1)^\top\,\mathbf{Mat}_i(\dot{A}(t))\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top,$$

which holds true if

$$\dot{\mathbf{S}} = \mathbf{U}(t_1)^\top\,\mathbf{Mat}_i(\dot{A}(t))\,\mathbf{V}(t_0) \qquad \text{and} \qquad \dot{\mathbf{U}} = \dot{\mathbf{V}} = \mathbf{0}.$$

Its solution after one time step $t_0 \to t_1$ is given by

$$\mathbf{S}(t_1) = \mathbf{U}(t_1)^\top\,\mathbf{Mat}_i(\Delta A)\,\mathbf{V}(t_0)$$

and therefore the solutions of the subproblems for Y_i^- for each $i = 1, \ldots, d$ are

$$\begin{aligned}Y_i^-(t_1) &= Y_i^-(t_0) - \mathrm{Ten}_i\big(\mathbf{U}(t_1)\,\mathbf{U}(t_1)^\top\,\mathbf{Mat}_i(\Delta A)\,\mathbf{V}(t_0)\,\mathbf{V}(t_0)^\top\big)\\ &= Y_i^-(t_0) - \mathrm{P}_{\mathbf{U}}\mathrm{P}_{\mathbf{V}}\Delta A\\ &= Y_i^-(t_0) - \mathrm{P}_{\mathbf{U}}\mathrm{P}_{\mathbf{W}}\Delta A\\ &= Y_i^-(t_0) - \mathrm{P}_i^-(Y_i^-)\Delta A.\end{aligned}$$

□

4.6.3 A direct exactness proof of the projector-splitting Tucker integrator

Though we can conclude by the equivalence result of the nested and the projector-splitting Tucker integrator, which will be shown in Section 4.6.5 that due to the exactness property of the nested Tucker integrator, the Tucker integrator from [Lub15] is also exact, we will give here a direct proof of the exactness property of the alternative Tucker integrator.

Theorem 4.5. *Suppose $A(t) \in \mathcal{M}$ is a low-rank tensor for all times $t_0 \le t \le T$ and $A(t_0) = Y(t_0)$. Then, the Tucker integrator of [Lub15] presented in Section 4.6.1 is exact, i.e. $Y^1 = A(t_1)$.*

A substantial ingredient for proving this theorem is the following key lemma.

Lemma 4.6. *Under the assumptions of Theorem 4.5, the solutions for $i = 1, \dots, d$ for the first $2d$ subproblems determined by the projector-splitting Tucker integrator are given as*

$$Y_i^+(t_1) = \mathrm{P}_{\mathbf{W}_i} A(t_1), \qquad \text{with} \qquad \mathbf{W}_i = \left(\mathrm{P}_{\mathbf{U}_{<i}(t_1)} \otimes \mathbf{I}_{n_{>i}}\right) \mathbf{U}_{\neq i}(t_0)\, \mathbf{Mat}_i(C_i(t_0))^\top$$
$$Y_i^-(t_1) = \mathrm{P}_{\mathbf{U}_{\leq i}(t_1)} A(t_0),$$

where

$$\mathrm{P}_{\mathbf{U}_{<i}(t_1)} = \bigotimes_{k=1}^{i-1} \mathrm{P}_{\mathbf{U}_k(t_1)}, \qquad \mathbf{I}_{n_{>i}} = \bigotimes_{k=i+1}^{d} \mathbf{I}_{n_k}, \qquad \mathbf{U}_{\neq i}(t_0) = \bigotimes_{\substack{l=1 \\ l \neq i}}^{d} \mathbf{U}_l(t_0).$$

Proof. The proof follows an induction over the modes $i = 1, \dots, d$. First, we show the result for both subproblems within mode 1.
For 1^+: Following Lemma 4.4, the solution of the first subproblem is given by

$$Y_1^+(t_1) = Y_1^+(t_0) + \mathrm{P}_1^+(Y_1^+(t_0))A(t_1) - \mathrm{P}_1^+(Y_1^+(t_0))A(t_0).$$

Since $Y(t_0) \in \mathcal{M}$, the initial value $Y_1^+(t_0) = Y(t_0)$ in matricized form is

$$\mathbf{Mat}_1(Y_1^+(t_0)) = \mathbf{Mat}_1(Y(t_0)) = \mathbf{U}_1(t_0)\mathbf{S}_1(t_0)\, \mathbf{V}_1(t_0)^\top.$$

Then, by definition of the orthogonal projection P_1^+, we find

$$\mathrm{P}_1^+(Y_1^+(t_0))A(t_1) = \mathrm{P}_1^+(Y(t_0))A(t_1) = \mathrm{P}_{\mathbf{V}_1(t_0)} A(t_1).$$

We observe that $\mathbf{S}_1(t_0)\, \mathbf{V}_1(t_0)^\top = \mathbf{Mat}_1(C_1(t_0))\, \mathbf{U}_{\neq 1}(t_0) = \mathbf{W}_1^\top$ and it follows that

$$\operatorname{span}(\mathbf{V}_1(t_0)) = \operatorname{span}(\mathbf{W}_1), \tag{4.47}$$

which is why

$$\mathrm{P}_1^+(Y_1^+(t_0))A(t_1) = \mathrm{P}_{\mathbf{W}_1} A(t_1).$$

Further, due to $A(t_0) = Y(t_0)$, we obtain

$$\mathrm{P}_1^+(Y_1^+(t_0))A(t_0) = \mathrm{P}_{\mathbf{V}_1(t_0)} A(t_0) = \mathrm{P}_{\mathbf{V}_1(t_0)} Y(t_0) = Y(t_0) = Y_1^+(t_0)$$

and therefore $Y_1^+(t_1) = \mathrm{P}_{\mathbf{W}_1} A(t_1)$.

For 1^-, the solution in closed form is given by

$$Y_1^-(t_1) = Y_1^-(t_0) - \mathrm{P}_1^-(Y_1^-(t_0))A(t_1) + \mathrm{P}_1^-(Y_1^-(t_0))A(t_0).$$

Now, since $A(t) \in \mathcal{M}$ for all $t_0 \leq t \leq T$, we can factorize its 1-mode matricization in SVD-like form, i.e., $\mathbf{Mat}_1(A(t)) = \mathbf{U}_{A,1}(t)\, \mathbf{Mat}_1(C_{A,1}(t))\big(\mathbf{U}_{A,2}(t)^\top \otimes \cdots \otimes \mathbf{U}_{A,d}(t)^\top\big)$. From the 1^+-step before we can write the initial value as

$$Y_1^-(t_0) = Y_1^+(t_1) = \mathrm{P}_{\mathbf{W}_1} A(t_1) = \mathrm{Ten}_1(\mathbf{Mat}_1(A(t_1))\, \mathbf{W}_1\, \mathbf{W}_1^\top).$$

Then, by the definition of the projection P_1^-, using (4.47) and since $A(t_0) = Y(t_0)$, we find

$$\begin{aligned}\mathrm{P}_1^-(Y_1^-(t_0))A(t_0) &= \mathrm{P}_1^-(Y_1^+(t_1))A(t_0) = \mathrm{P}_{\mathbf{U}_{A,1}(t_1)}\mathrm{P}_{\mathbf{W}_1}A(t_0)\\ &= \mathrm{P}_{\mathbf{U}_{A,1}(t_1)}\mathrm{P}_{\mathbf{V}_1(t_0)}Y(t_0) = \mathrm{P}_{\mathbf{U}_{A,1}(t_1)}Y(t_0)\\ &= \mathrm{P}_{\mathbf{U}_{A,1}(t_1)}A(t_0),\end{aligned} \tag{4.48}$$

where $\mathrm{P}_{\mathbf{V}_1(t_0)}$ and $\mathrm{P}_{\mathbf{U}_{A,1}(t_1)}$ are defined analogously as $\mathrm{P}_{\mathbf{V}}$ and $\mathrm{P}_{\mathbf{U}}$ in (4.46). The solution of the first substep for 1^+ of the alternative integrator yields an intermediate approximation tensor with updated basis matrix $\mathbf{U}_1$ in the first mode and a new matrix $\mathbf{S}_1$, i.e., we can also write the initial value of the current substep as $Y_1^-(t_0) = Y_1^+(t_1) = \mathrm{Ten}_1(\mathbf{U}_1(t_1)\mathbf{S}_1(t_1)\,\mathbf{V}_1(t_0)^\top)$. Hence, with (4.47), we also find

$$\mathrm{P}_1^-(Y_1^-(t_0))A(t_0) = \mathrm{P}_{\mathbf{U}_1(t_1)}\mathrm{P}_{\mathbf{W}_1}A(t_0) = \mathrm{P}_{\mathbf{U}_1(t_1)}A(t_0). \tag{4.49}$$

Comparing this with (4.48), it follows that

$$\mathrm{P}_{\mathbf{U}_{A,1}(t_1)} = \mathrm{P}_{\mathbf{U}_1(t_1)} \tag{4.50}$$

and so $\mathrm{P}_1^-(Y_1^-(t_0))A(t_0) = \mathrm{P}_{\mathbf{U}_1(t_1)}A(t_0)$. Further, by using the result of the 1^+-step, we have

$$\begin{aligned}\mathrm{P}_1^-(Y_1^-(t_0))A(t_1) &= \mathrm{P}_1^-(Y_1^+(t_1))A(t_1) = \mathrm{P}_{\mathbf{U}_{A,1}(t_1)}\mathrm{P}_{\mathbf{W}_1}A(t_1) = \mathrm{P}_{\mathbf{W}_1}A(t_1) = Y_1^+(t_1)\\ &= Y_1^-(t_0)\end{aligned}$$

and therefore $Y_1^-(t_1) = \mathrm{P}_{\mathbf{U}_1(t_1)}A(t_0)$.

The results for the next substeps are shown by an inductive argument. Suppose, this has been shown for all substeps $2^\pm, \dots, (i-1)^\pm$. Then, following Lemma 4.4, the solution for i^+ is given as

$$Y_i^+(t_1) = Y_i^+(t_0) + \mathrm{P}_i^+(Y_i^+(t_0))A(t_1) - \mathrm{P}_i^+(Y_i^+(t_0))A(t_0).$$

Using the result from the previous $(i-1)^-$-step and since by assumption $A(t_0) = Y(t_0)$, we can write the initial value of the current subproblem as

$$\begin{aligned}Y_i^+(t_0) &= Y_{i-1}^-(t_1) + \mathrm{P}_{\mathbf{U}_{\le i-1}(t_1)}A(t_0) = \mathrm{P}_{\mathbf{U}_{\le i-1}(t_1)}Y(t_0)\\ &= \mathrm{Ten}_i\Big(\mathbf{U}_i(t_0)\,\mathbf{Mat}_i(C_i(t_0))\big(\mathbf{U}_{\neq i}(t_0)^\top(\mathrm{P}_{\mathbf{U}_{<i}(t_1)} \otimes \mathbf{I}_{n_{>i}})\big)\Big)\\ &= \mathrm{Ten}_i\big(\mathbf{U}_i(t_0)\,\mathbf{W}_i^\top\big).\end{aligned} \tag{4.51}$$

Hence, by the definition of the projection P_i^+, we conclude by Lemma 4.4 that this projection is given as

$$\mathrm{P}_i^+(Y_i^+(t_0))A(t) = \mathrm{P}_{\mathbf{W}_i}A(t), \qquad \text{for all } t, \tag{4.52}$$

i.e. in particular for $t = t_1$ and so $\mathrm{P}_i^+(Y_i^+(t_0))A(t_1) = \mathrm{P}_{\mathbf{W}_i}A(t_1)$. Taking a closer look on the definition of $\mathbf{W}_i$, we observe that

$$\mathrm{span}(\mathbf{W}_i) \subset \mathrm{span}(\mathrm{P}_{\mathbf{U}_{\le i-1}(t_1)})$$

and hence $\mathrm{P}_{\mathbf{W}_i} = \mathrm{P}_{\mathbf{W}_i}\mathrm{P}_{\mathbf{U}_{\leq i-1}(t_1)}$. Using this property of the projection, applying (4.52) and the form (4.51) of the initial value, we obtain for the projected initial value

$$\begin{aligned}\mathrm{P}_i^+(Y_i^+(t_0))A(t_0) &= \mathrm{P}_{\mathbf{W}_i}A(t_0) = \mathrm{P}_{\mathbf{W}_i}Y(t_0) = \mathrm{P}_{\mathbf{W}_i}\mathrm{P}_{\mathbf{U}_{\leq i-1}(t_1)}Y(t_0)\\ &= \mathrm{P}_{\mathbf{W}_i}\,\mathrm{Ten}_i\big(\mathbf{U}_i(t_0)\,\mathbf{W}_i^\top\big) = \mathrm{Ten}_i\big(\mathbf{U}_i(t_0)\,\mathbf{W}_i^\top\big)\\ &= Y_i^+(t_0).\end{aligned}$$

Therefore, $Y_i^+(t_1) = \mathrm{P}_{\mathbf{W}_i}A(t_1)$.

For the second substep of the integrator within this mode, we now consider the solution to the i^--subproblem, which is given by

$$Y_i^-(t_1) = Y_i^-(t_0) - \mathrm{P}_i^-(Y_i^-(t_0))A(t_1) + \mathrm{P}_i^-(Y_i^-(t_0))A(t_0).$$

By assumption, $A(t) \in \mathcal{M}$ for all times $t_0 \leq t \leq T$, and so we can determine an SVD-like factorization of its i-mode matricization, i.e.,

$$\mathbf{Mat}_i(A(t)) = \mathbf{U}_{A,i}(t)\,\mathbf{Mat}_i(C_{A,i}(t))\Big(\bigotimes_{\substack{l=1\\l\neq i}}^{d}\mathbf{U}_{A,l}(t)^\top\Big).$$

With this factorization at hand, we can rewrite the initial value of the current subproblem as

$$Y_i^-(t_0) = Y_{i-1}^+(t_1) = \mathrm{P}_{\mathbf{W}_i}A(t_1) = \mathrm{Ten}_i\Big(\mathbf{U}_{A,i}(t_1)\,\mathbf{Mat}_i(C_{A,i}(t_1))\Big(\bigotimes_{\substack{l=1\\l\neq i}}^{d}\mathbf{U}_{A,l}(t_1)^\top\Big)\,\mathbf{W}_i\,\mathbf{W}_i^\top\Big). \tag{4.53}$$

Now, after having computed the previous substep within the integration scheme, we can also write the initial value in terms of its matricized solution as

$$Y_i^-(t_0) = Y_i^+(t_1) = \mathrm{Ten}_i(\mathbf{U}_i(t_1)\mathbf{S}_i(t_1)\,\mathbf{V}_i^{0,\top}),$$

with $\mathbf{V}_i$ as defined within the integrator, see, e.g., line 4 in Algorithm 7 in Section 4.6.1. A comparison of those two forms of the initial value of the current subproblem implies that the columns of $\mathbf{U}_{A,i}(t_1)$ and the columns of $\mathbf{U}_i(t_1)$ span the same subspace of $\mathcal{M}$, since both constitute the range of the initial value $Y_i^-(t_0)$, i.e.,

$$\mathrm{P}_{\mathbf{U}_{A,i}(t_1)} = \mathrm{P}_{\mathbf{U}_i(t_1)}. \tag{4.54}$$

From the previous i^+-step, we know that $\mathrm{P}_{\mathbf{W}_i}A(t_0) = Y_i^+(t_0)$ and $Y_i^+(t_0) = Y_{i-1}^-(t_1)$, which again by the result in the $(i-1)^-$-step equals $\mathrm{P}_{\mathbf{U}_{\leq i-1}(t_1)}A(t_0)$. Hence

$$\mathrm{P}_i^-(Y_i^-(t_0))A(t_0) = \mathrm{P}_{\mathbf{U}_i(t_1)}\mathrm{P}_{\mathbf{U}_{\leq i-1}(t_1)}A(t_0) = \mathrm{P}_{\mathbf{U}_{\leq i}(t_1)}A(t_0).$$

Futher, due to the result in the previous step, and the key observation (4.54), we obtain

$$\begin{aligned}\mathrm{P}_i^-(Y_i^-(t_0))A(t_1) &= \mathrm{P}_i^-(Y_i^+(t_1)) = \mathrm{P}_{\mathbf{U}_i(t_1)}\mathrm{P}_{\mathbf{W}_i}A(t_1) = \mathrm{P}_{\mathbf{U}_{A,i}(t_1)}\mathrm{P}_{\mathbf{W}_i}A(t_1)\\ &= \mathrm{P}_{\mathbf{W}_i}A(t_1) = Y_i^+(t_1)\\ &= Y_i^-(t_0).\end{aligned}$$

Therefore, we conclude $Y_i^-(t_1) = \mathrm{P}_{\mathbf{U}_{\leq i}(t_1)}A(t_0)$. □

With this essential lemma at hand, we are now in the position to prove the theorem about the exactness of the projector-splitting Tucker integrator provided in [Lub15] and recalled in Section 4.6.1.

Proof of Theorem 4.5. In the previous lemma we have seen the form of the solutions to the subproblems of the integration scheme and in particular how to update the basis matrices $\mathbf{U}_i$ for all $i = 1, \dots, d$. In order to show exactness of the projector-splitting Tucker integrator, we are left with showing the update of the core tensor. To this end, we consider its differential equation, which is given as

$$\dot{Y}_C(t) = \dot{A}(t) \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top.$$

Following the idea of a splitting integrator, we take the last updated substep as initial value for solving the above differential equation, such that Lemma 4.6 gives

$$\begin{aligned}
Y^1 &= \mathrm{Ten}_d(Y_d^-(t_1)) + \Delta A \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top \\
&= \mathrm{P}_{\mathbf{U}_{\leq d}(t_1)} A(t_0) + A(t_1) \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top - A(t_0) \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top \\
&= A(t_0) \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top + A(t_1) \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top - A(t_0) \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top \\
&= A(t_1) \bigtimes_{i=1}^{d} \mathbf{U}_i(t_1)\mathbf{U}_i(t_1)^\top \\
&= \mathrm{P}_{\mathbf{U}_{\leq d}(t_1)} A(t_1) \\
&= A(t_1),
\end{aligned}$$

where in the last equality we have used the key observation (4.54) that the projection onto the space spanned by the columns of $\mathbf{U}_{A,i}(t_1)$ is the same as the projection onto the column space of $\mathbf{U}_i(t_1)$ for all $i = 1, \dots, d$. □

4.6.4 Discussion and comparison

The derivation of the nested Tucker integrator presented in Section 4.2 is based on the idea of solving the differential equations for $\mathbf{K}_i$ and $\mathbf{S}_i$ directly, but computing a low-rank approximation of the ODE for $\mathbf{L}_i$, where $i = 1, \dots, d-1$. This is a conceptually different derivation from the time integrator described in Section 4.6.1.

In this approach we do not consider a differential equation for the matrix $\mathbf{L}(t)$ in each modal step, but compute an update of the core tensor $C(t)$ by solving the corresponding tensor differential equation at the end of the integration steps after having updated the matrices $\overline{\mathbf{K}}_i$ and $\overline{\mathbf{S}}_i$ for each mode $i = 1, \dots, d$. A schematic illustration is given in Figure 4.7.

Comparing Algorithm 7 with Algorithm 6 for the nested Tucker integrator, we see that the two algorithms solve different matrix differential equations for $\mathbf{K}_i(t)$ and $\mathbf{S}_i(t)$. The reason is the positioning of the updated basis matrices $\mathbf{U}_i^1$: in Algorithm 7, the corange $\mathbf{V}_i^{0,\top}$ of the current unfolded approximation tensor takes those basis matrices after

performing one time step, whereas in Algorithm 6, they are provided on the right-hand side of the differential equation for $Y_i(t)$. Therefore, we observe that the time integrator described in [Lub15] does not reduce the dimension in each mode, contrary to the nested Tucker integrator, which diminishes the dimension of the current mode due to the inexact solution in the third substeps for each mode $i = 1, \ldots, d-1$.

In addition, the nested Tucker integrator solves a matrix differential equation for $\mathbf{L}(t)$ and the Tucker integrator proposed in [Lub15] deals with solving a tensor differential equation for updating the core tensor $C(t)$. So from the computational point of view, the latter is more expensive.

We illustrate the integration method of [Lub15] in Figure 4.7 for ease of comparison with Figure 4.6.

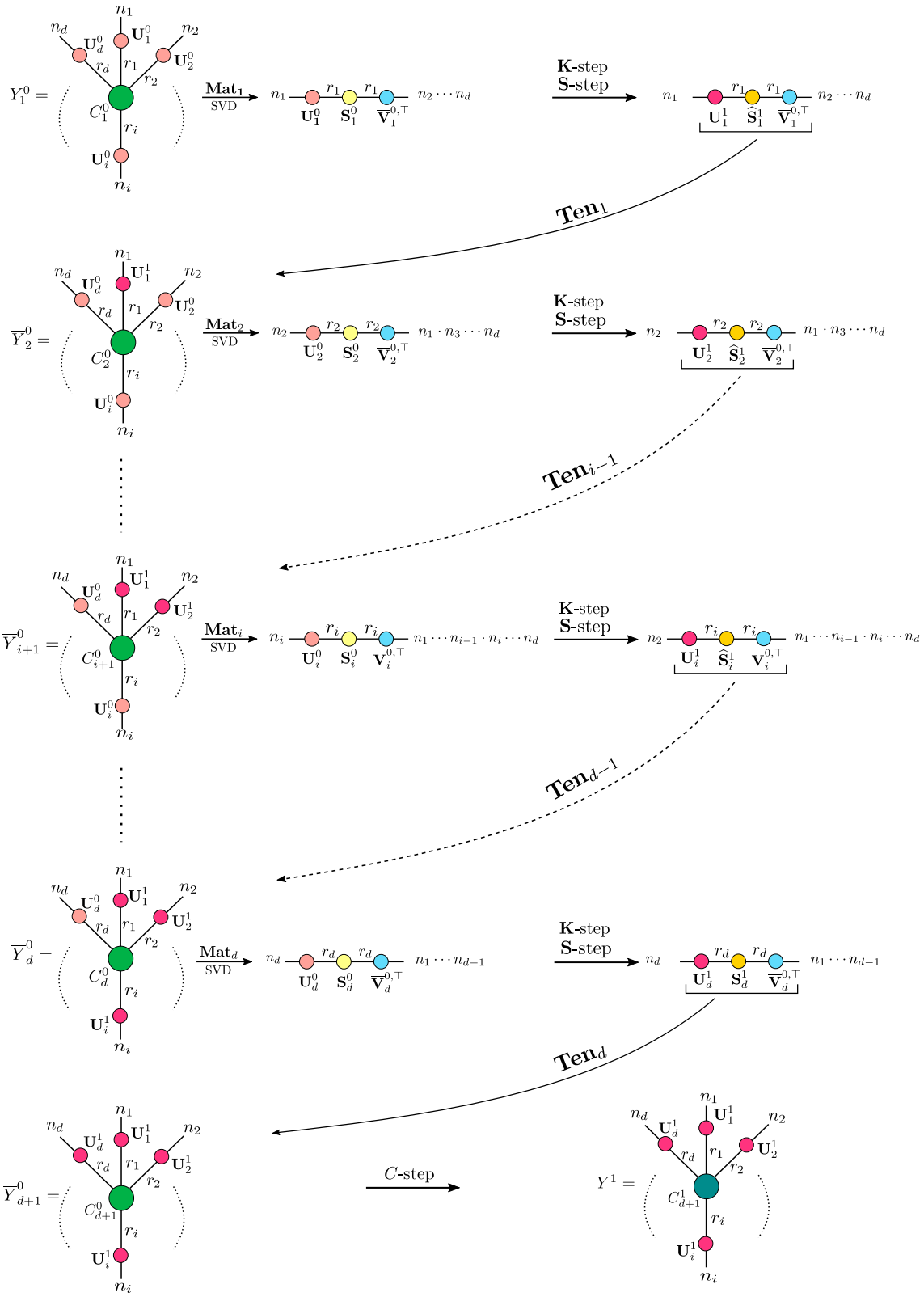

Figure 4.7: Illustration of the Tucker integrator proposed in [Lub15].

4.6.5 Mathematical equivalence

Going through the lines of Algorithms 6 and 7, we clearly see that the main differences are the matrix differential equations that have to be solved. Hence, it is sufficient to show equivalence of the matrix differential equations that appear in Algorithms 6 and 7. In order to distinguish between the factors computed by those two methods, we will follow the notation used in the description of the integrator in Section 4.6.1 and denote those for Algorithm 7 using $\overline{\cdot}$, e.g., $\overline{\mathbf{V}}_i$, $\overline{\mathbf{K}}_i(t)$, and $\overline{\mathbf{S}}_i(t)$. The notation for Algorithm 6 is left unchanged.

Theorem 4.7. *The nested Tucker integrator presented in Section 4.2 (Algorithm 6) and the Tucker integrator described in [Lub15] (Algorithm 7) applied on the tensor differential equation* (4.1) *are equivalent in the sense that they yield the same low-rank approximation after each time step.*

We will show the proof for the first time step from $t_0 \to t_1$. The same arguments hold for the subsequent time steps, where we take the result from the previous one as initial value, such that the following proof holds for each time step.

Proof. We start with Algorithm 7. Writing line 4 as

$$\overline{\mathbf{V}}_i^{0,\top} = \mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top} \otimes \bigotimes_{k=i+1}^{d} \mathbf{U}_k^{0,\top}\Big),$$

the equation of motion of $\mathbf{K}_i$ becomes

$$\begin{aligned}\dot{\overline{\mathbf{K}}}_i(t) &= \mathbf{Mat}_i\Big(F\big(t, \mathrm{Ten}_i(\overline{\mathbf{K}}_i(t)\overline{\mathbf{V}}_i^{0,\top})\big)\Big)\overline{\mathbf{V}}_i^0 \\ &= \mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i\Big(\overline{\mathbf{K}}_i(t)\,\mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top} \otimes \bigotimes_{k=i+1}^{d} \mathbf{U}_k^{0,\top}\Big)\Big)\Big)\Big)\overline{\mathbf{V}}_i^0.\end{aligned}$$

For Algorithm 6, on the other hand that equation reads

$$\dot{\mathbf{K}}_i(t) = \mathbf{Mat}_i\Big(F\Big(t, \mathrm{Ten}_i(\mathbf{K}_i(t)\,\mathbf{V}_i^{0,\top}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1\Big) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_i^0\,.$$

We first expand the argument of F in this ODE. Writing line 4 in Algorithm 6 as

$$\mathbf{V}_i^{0,\top} = \mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=1}^{i-1} \mathbf{I}_{r_k} \otimes \bigotimes_{k=i+1}^{d} \mathbf{U}_k^{0,\top}\Big) \tag{4.55}$$

and substituting, we obtain

$$\begin{aligned}\mathrm{Ten}_i(\mathbf{K}_i(t)\,\mathbf{V}_i^{0,\top}) \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 &= \mathrm{Ten}_i(\mathbf{K}_i(t)\,\mathbf{Q}_i^{0,\top}) \bigtimes_{k=i+1}^{d} \mathbf{U}_k^0 \bigtimes_{k=1}^{i-1} \mathbf{U}_k^1 \\ &= \mathrm{Ten}_i\Big(\mathbf{K}_i(t)\,\mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=i+1}^{d} \mathbf{U}_k^{0,\top} \otimes \bigotimes_{k=1}^{i-1} \mathbf{U}_k^{1,\top}\Big)\Big).\end{aligned}$$

Hence, we see that F has the same arguments in both algorithms for the $\mathbf{K}$-step. For ease of clarity, we omit the argument and continue with

$$\begin{aligned}\dot{\mathbf{K}}_i(t) &= \mathbf{Mat}_i\Big(F(t,\cdot)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_i^0\\ &= \mathbf{Mat}_i\Big(F(t,\cdot)\Big)\Big(\bigotimes_{k=1}^{i-1}\mathbf{U}_k^1\otimes\bigotimes_{k=i+1}^{d}\mathbf{I}_{r_k}\Big)\Big(\bigotimes_{k=1}^{i-1}\mathbf{I}_{r_k}\otimes\bigotimes_{k=i+1}^{d}\mathbf{U}_k^0\Big)\,\mathbf{Q}_i^0\\ &= \mathbf{Mat}_i\Big(F(t,\cdot)\Big)\Big(\bigotimes_{k=1}^{i-1}\mathbf{U}_k^1\otimes\bigotimes_{k=i+1}^{d}\mathbf{U}_k^0\Big)\,\mathbf{Q}_i^0\,.\end{aligned}$$

Comparing with $\overline{\mathbf{V}}_i^0$ above, we see that the differential equations for $\mathbf{K}_i(t)$ and $\overline{\mathbf{K}}_i(t)$ are indeed equivalent.

Hence, applying the same numerical method to both of them would give the same result $\mathbf{K}_i^1 = \overline{\mathbf{K}}_i^1$.

The equivalence of the evolution equations for $\dot{\mathbf{S}}_i(t)$ and $\dot{\overline{\mathbf{S}}}_i(t)$ can be shown in a similar way as above. With $\overline{\mathbf{V}}_i^{0,\top}$ from line 4, line 10 in Algorithm 7 reads

$$\begin{aligned}\dot{\overline{\mathbf{S}}}_i(t) &= -\,\mathbf{U}_i^{1,\top}\,\mathbf{Mat}_i\Big(F\big(t,\mathrm{Ten}_i(\mathbf{U}_i^1\,\overline{\mathbf{S}}_i(t)\overline{\mathbf{V}}_i^{0,\top})\big)\Big)\overline{\mathbf{V}}_i^0\\ &= -\,\mathbf{U}_i^{1,\top}\,\mathbf{Mat}_i\Big(F\Big(t,\mathrm{Ten}_i\Big(\mathbf{U}_i^1\,\overline{\mathbf{S}}_i(t)\,\mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\otimes\bigotimes_{k=i+1}^{d}\mathbf{U}_k^{0,\top}\Big)\Big)\Big)\Big)\overline{\mathbf{V}}_i^0\\ &= -\,\mathbf{U}_i^{1,\top}\,\mathbf{Mat}_i\Big(F\Big(t,\mathrm{Ten}_i\Big(\overline{\mathbf{S}}_i(t)\,\mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=1}^{i}\mathbf{U}_k^{1,\top}\otimes\bigotimes_{k=i+1}^{d}\mathbf{U}_k^{0,\top}\Big)\Big)\Big)\Big)\overline{\mathbf{V}}_i^0.\end{aligned}\tag{4.56}$$

In Algorithm 6 this line is

$$\dot{\mathbf{S}}_i(t) = -\,\mathbf{U}_i^{1,\top}\,\mathbf{Mat}_i\Big(F\Big(t,\mathrm{Ten}_i(\mathbf{U}_i^1\,\mathbf{S}_i(t)\,\mathbf{V}_i^{0,\top})\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1\Big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^{1,\top}\Big)\,\mathbf{V}_i^0\,.$$

With the same $\overline{\mathbf{V}}_i^{0,\top}$ as given in (4.55), we have for the argument of F within Algorithm 6:

$$\begin{aligned}\mathrm{Ten}_i(\mathbf{U}_i^1\,\mathbf{S}_i(t)\,\mathbf{V}_i^{0,\top})\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1 &= \mathrm{Ten}_i\Big(\mathbf{U}_i^1\,\mathbf{S}_i(t)\,\mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=1}^{i-1}\mathbf{I}_{r_k}\otimes\bigotimes_{k=i+1}^{d}\mathbf{U}_k^{0,\top}\Big)\Big)\bigtimes_{k=1}^{i-1}\mathbf{U}_k^1\\ &= \mathrm{Ten}_i(\mathbf{S}_i(t)\,\mathbf{Q}_i^{0,\top})\bigtimes_{k=i+1}^{d}\mathbf{U}_k^0\bigtimes_{k=1}^{i}\mathbf{U}_k^1\\ &= \mathrm{Ten}_i\Big(\mathbf{S}_i(t)\,\mathbf{Q}_i^{0,\top}\Big(\bigotimes_{k=i+1}^{d}\mathbf{U}_k^{0,\top}\otimes\bigotimes_{k=1}^{i}\mathbf{U}_k^{1,\top}\Big)\Big).\end{aligned}$$

Comparing this argument with the one in line 10 of Algorithm 7 or rather in (4.56), we observe that F has the same argument.

Since $\mathbf{U}_i^1$ as the first factor in both differential equations, the one for $\overline{\mathbf{S}}(t)$ and the one for $\mathbf{S}(t)$, is the same, and we have already seen equivalence regarding the multiplication by $\overline{\mathbf{V}}_i^0$ or $\mathbf{V}_i^0$ in the ODEs for $\overline{\mathbf{K}}(t)$ and $\mathbf{K}(t)$, respectively, we conclude the equivalence of $\overline{\mathbf{S}}(t)$ and $\mathbf{S}(t)$. This gives the same numerical solutions after one time step, i.e., $\mathbf{S}_i^1 = \overline{\mathbf{S}}_i^1$.

Finally, for the core tensor, we compare the differential equation for $C(t)$ in Algorithm 7,

$$\dot{C}(t) = F\big(t, C(t) \bigtimes_{i=1}^{d} \mathbf{U}_i^1\big) \bigtimes_{i=1}^{d} \mathbf{U}_i^{1,\top}, \qquad C(t_0) = C^0,$$

with that of $\mathbf{L}(t)$ from Algorithm 6. Retensorizing the latter in the dth mode yields

$$\begin{aligned}
\mathrm{Ten}_d(\dot{\mathbf{L}}(t)^\top) &= \mathrm{Ten}_d\Big(\mathbf{U}_d^{1,\top}\, \mathbf{Mat}_d\Big(F\Big(t, \mathrm{Ten}_d(\mathbf{U}_d^1\, \mathbf{L}(t)^\top) \bigtimes_{i=1}^{d-1} \mathbf{U}_i^1\Big) \bigtimes_{i=1}^{d-1} \mathbf{U}_i^{1,\top}\Big)\Big) \\
&= \mathrm{Ten}_d\Big(\mathbf{U}_d^{1,\top}\, \mathbf{Mat}_d\Big(F\Big(t, \mathrm{Ten}_d(\mathbf{L}(t)^\top) \bigtimes_{i=1}^{d} \mathbf{U}_i^1\Big) \bigtimes_{i=1}^{d-1} \mathbf{U}_i^{1,\top}\Big)\Big) \\
&= F\big(t, \mathrm{Ten}_d(\mathbf{L}(t)^\top) \bigtimes_{i=1}^{d} \mathbf{U}_i^1\big) \bigtimes_{i=1}^{d} \mathbf{U}_i^{1,\top}.
\end{aligned}$$

Identifying now $C(t)$ as $\mathrm{Ten}_d(\mathbf{L}(t)^\top)$, we see that the differential equations are the same. Since this also holds true for the initial values,

$$\mathrm{Ten}_d(\mathbf{L}_d^{0,\top}) = \mathrm{Ten}_d(\mathbf{Mat}_d(C^0)) = C^0,$$

both algorithms deliver the same low-rank approximation Y^1. □

4.7 Numerical experiments

We present two numerical examples to illustrate our theoretical results of the proposed nested Tucker integrator. We consider examples that are tensor variants of the examples in [KLW16, Section 4] for the matrix case and are taken from [LVW18], in particular, approximately adding tensors and an example of a discrete nonlinear Schrödinger equation.

4.7.1 Approximate addition of tensors

Let $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ be a tensor of multilinear rank $r = (r_1, \ldots, r_d)$ and let $B \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. We consider the addition of the two given tensors, which results in

$$C = A + B, \tag{4.57}$$

where C typically is not of low rank. We aim to find an approximation tensor of multilinear rank $(r_1, \ldots, r_d)$. Such a computation is for example required in optimization problems on low-rank manifolds, under the name of retractions, and need to be computed in each iterative step, see [AO15]. There, the increment is typically a tangential tensor $B \in \mathcal{T}_A\mathcal{M}$, which after adding directly as in (4.57) yields a tensor C of multilinear rank $(2r_1, \ldots, 2r_d)$. Afterwards, the result is projected onto $\mathcal{M}$ by an SVD-based rank r approximation in order to obtain an approximation tensor $\widetilde{Y}^1 \in \mathcal{M}$. With this procedure, we first leave the low-rank manifold and then project back onto $\mathcal{M}$.

Instead, we propose to apply one time step of the nested Tucker integrator starting with $t_0 = 0$ and with time step size $h = 1$ in order to solve

$$\dot{Y}(t) = \mathrm{P}(Y)B, \qquad Y(t_0) = A. \tag{4.58}$$

This gives an approximate solution $Y^1 \in \mathcal{M}$ for the result of the direct addition (4.57). Contrary to the standard approach, we never leave the low-rank manifold when applying the nested Tucker integrator.

Note that in case when $\mathrm{P}(Y)$ is the identity map, the differential equation (4.58) simplifies to $\dot{Y}(t) = B,\ Y(t_0) = A$. Solving this ODE for one time step $t_0 \to t_0 + h$ with $h = 1$, we obtain $Y(1) = A + B$. Hence solving this ODE yields the same result as the direct addition (4.57). The approximation tensor $Y(1)$ is not of low rank, since we have not projected the tensor B onto the tangent space of the low-rank manifold. Due to this drawback, this is not our method of choice.

For our numerical example, we initialize A as a random Tucker tensor of size $100 \times 100 \times 100$ and multilinear rank $r = (10, 10, 10)$. The increment B is constructed to be a random tensor in the tangent space $\mathcal{T}_A\mathcal{M}$. We compare the full rank addition (4.57) with the low-rank approximation $Y^1 \in \mathcal{M}$ obtained by the nested Tucker integrator. We also compare those results with the retracted rank $2r$ approximation $\widetilde{Y}^1$, for which we perform a best rank r approximation. The figure below illustrates those comparisons:

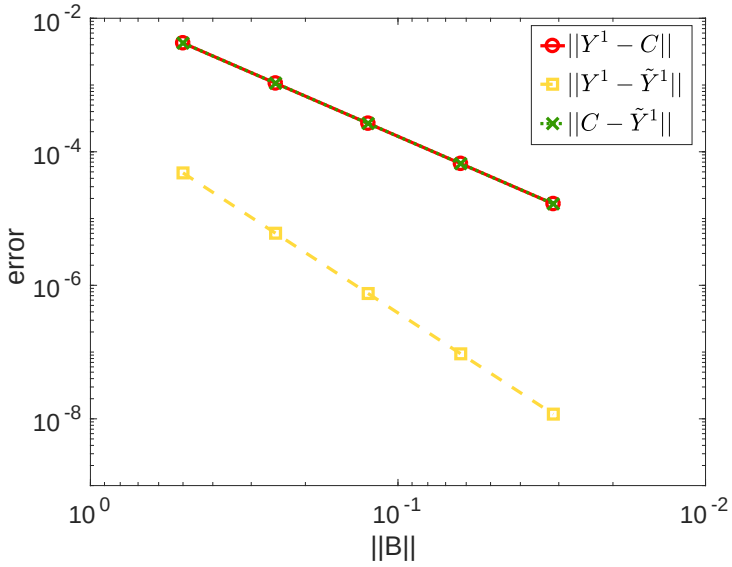

Figure 4.8: Errors for tensor addition for tangential increments B of decreasing norm.

We observe that the errors decrease with decreasing norm of the increment tensor B. We also see that the difference between the errors of the splitting integrator and the projected direct addition is marginal.

4.7.2 A discrete nonlinear Schrödinger equation for tensors

We choose an example, where the differential equation consists of a linear and a nonlinear part, where the nonlinearity is controlled by ε, such that we can see the error behavior of the nested Tucker integrator.

To this end, we model a dilute Bose–Einstein condensate, trapped in a periodic po-

tential, see [TS01], on a regular lattice of width γ. The dynamics of its phase diagram is governed by the discrete nonlinear Schrödinger equation

$$\begin{aligned} i\dot{A}(t) &= -\frac{1}{2}L[A(t)] + \varepsilon|A(t)|^2 \odot A(t) \\ A_{jkl}(t_0) &= \exp\bigl(-1/\gamma^2((j-j_1)^2-(k-k_1)^2-(l-l_1)^2)\bigr) \\ &\quad + \exp\bigl(-1/\gamma^2((j-j_2)^2-(k-k_2)^2-(l-l_2)^2)\bigr), \end{aligned} \tag{4.59}$$

where $A(t) \in \mathbb{R}^{n_1\times n_2\times n_3}$ with $n_i = 100$ for all $i \in \{1,2,3\}$ and $\odot$ denotes the entrywise (Hadamard) product. The bounded linear operator $L\colon \mathbb{R}^{n_1\times n_2\times n_3} \to \mathbb{R}^{n_1\times n_2\times n_3}$ describes the interaction between the grid points centered at (j,k,l) for all $j,k,l = 1,\dots,100$. It is defined componentwise as

$$\begin{aligned} L[A](j,k,l) = {} & A(j-1,k,l) + A(j+1,k,l) + A(j,k-1,l) + A(j,k+1,l) \\ & + A(j,k,l-1) + A(j,k,l+1), \end{aligned}$$

where terms with indices outside the range from 1 to 100 are interpreted as 0. Compared to the more standard seven-point stencil for the discrete Laplace operator in three dimensions, the operator L does not take the centered grid point into account. The entries of the tensor $|A|^2$ are the squares of the absolute values of the corresponding entries of A. The parameter ε determines the degree of nonlinearity.

We consider two excitations of the system, which are located at grid-points $(j_1,k_1,l_1) = (75,25,1)$ and $(j_2,k_2,l_2) = (25,75,100)$. We take $\gamma = 10$.

To compute a low-rank approximation $Y(t) \in \mathcal{M}$ with multilinear rank $r = (10,10,10)$ to the solution of the nonlinear differential equation (4.59), we apply the nested Tucker integrator (Algorithm 6) to (4.59). The differential equations appearing in the substeps of each mode are solved by the classical 4th-order Runge–Kutta method with sub time step size $h = 10^{-3}$. This approximate solution is compared to a full rank reference solution, which is also computed by a 4th-order Runge–Kutta method, but with $h = 0.5\cdot 10^{-3}$. In Table 4.1 we show the error behavior for different parameters ε and time step sizes h:

$\varepsilon \setminus h$	1	10^{-1}	10^{-2}	10^{-3}
1	4.59e-1	4.01e-2	3.88e-2	3.88e-2
10^{-1}	9.39e-2	9.68e-4	1.61e-4	1.47e-4
10^{-2}	9.27e-3	3.20e-5	2.19e-6	1.30e-6
10^{-3}	5.36e-4	3.18e-6	8.93e-8	3.54e-8
10^{-4}	5.12e-5	2.73e-7	3.23e-9	1.91e-9

Table 4.1: Error in Frobenius norm at $t = 1$ of the rank (10,10,10) nested Tucker integrator applied to (4.59).

For each time step size h, we see the error decaying with ε. This observation is due to the fact that the linear term $L[A(t)]$ in (4.59) maps onto the tangent space $\mathcal{T}_Y\mathcal{M}$ of the

manifold $\mathcal{M}$ of multilinear rank. The nonlinear term is of full rank, but it is controlled by the factor ε. This makes the dependence of the error behavior on ε explicit. We also see in the first row that the error stagnates from time step size $h = 10^{-2}$ on and this shows the dominance of the perturbation factor ε. We would observe the same behavior for smaller ε, but for smaller time step sizes.

Finally, in the last row, where the influence of ε is small, we observe convergence of the error in terms of the time step size h of the order given in Theorem 4.3.

5 Further result and future research

Based on the nested Tucker integrator, we discuss two other tensor formats, whose numerical analysis and the integration method itself are related to the ideas in the Tucker case.

We are given the tensor differential equation

$$\dot{A}(t) = F(t, A(t)), \qquad A(t_0) = A^0, \tag{5.1}$$

where $A(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is the unknown solution, which we aim to approximate by a tensor of the same size, but of low rank. To this end, we follow the ansatz of the dynamical low-rank approximation described in Section 1.2 and evolve a differential equation for the approximation tensor $Y(t)$ of low rank.

In this chapter, we are concerned with an integration method in case when $Y(t)$ is in the tensor train format and we discuss its error behavior. In fact, the time integrator for tensor trains extends the matrix projector-splitting integrator. The link between the approaches for tensor trains and for Tucker tensors is that the derivation of the Tucker integrator presented in Section 4.2 can be traced back to the ideas used in the error analysis of the time integrator for tensor trains. The time integration of tensor trains described in this chapter is taken from [LOV15] and the error analysis is first proposed in [KLW16].

We also give a perspective for future research about tensor tree networks and their time integration. The ideas from the design and analysis of the nested Tucker integrator can be extended to the tensor tree network representation as a generalized low-rank format, which includes Tucker tensors and tensor trains.

5.1 Time integration of rank-constrained tensor trains

We start with recalling the *tensor train decomposition* (TT) introduced in [OT09, Ose11]. A tensor $Y(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is in the tensor train format if there exist core tensors $C_i(t) \in \mathbb{R}^{r_{i-1} \times n_i \times r_i}$ of full multilinear rank, such that every element of $Y(t)$ is of the form

$$Y(k_1, \ldots, k_d)(t) = \sum_{j_1=1}^{r_1} \cdots \sum_{j_{d-1}=1}^{r_{d-1}} C_1(1, k_1, j_1)(t) \cdot C_2(j_1, k_2, j_2)(t) \cdots C_d(j_{d-1}, k_d, 1)(t),$$

where $k_i = 1, \ldots, n_i$ and $i = 1, \ldots, d$. The rank of the tensor train $Y(t)$ is defined to be $r = (1, r_1, \ldots, r_{d-1}, 1) \in \mathbb{N}^{d+1}$. We aim to approximate the unknown solution of the tensor

differential equation (5.1) for $A(t)$ by a tensor $Y(t)$ that is in the low-rank manifold

$$\mathcal{M} := \{Y(t) \in \mathbb{R}^{n_1 \times \cdots \times n_d} | \operatorname{rank} Y(t) = (1, r_1, \ldots, r_{d-1}, 1)\},$$

which in fact is shown to be a manifold, see [HRS12, UV13]. To this end, we follow the ansatz of the dynamical low-rank approximation, see Section 1.2 for tensors instead for matrices (simply by exchanging the matrix $\mathbf{Y}(t)$ by the tensor $Y(t)$), and consider the reduced model equation

$$\dot{Y}(t) = \mathrm{P}(Y(t))F(t, Y(t)), \qquad Y(t_0) = Y^0 \in \mathcal{M}, \tag{5.2}$$

where $\mathrm{P}(Y(t))$ is the orthogonal projection onto the tangent space $\mathcal{T}_{Y(t)}\mathcal{M}$ of the low-rank manifold $\mathcal{M}$ at $Y(t)$.

The integration procedure proposed in [LOV15] is an extension of the matrix projector-splitting integrator described in Section 2.1. It is based on singular value decompositions of unfoldings of the approximation tensor, which are of the form (omitting the time dependence)

$$\mathbf{Unf}_i(Y) = \mathbf{U}_i \, \mathbf{S}_i \, \mathbf{V}_{i+1}^\top .$$

We refer to Section 4.1.2 for a recapitulation of how to unfold a tensor into a matrix. The singular value decomposition of the unfoldings of Y can be obtained recursively in both directions, i.e., starting from the SVD of $\mathbf{Unf}_i(Y)$, we can determine the SVD of $\mathbf{Unf}_{i-1}(Y)$ as well as of $\mathbf{Unf}_{i+1}(Y)$, respectively.

Using this factorization, the orthogonal projection in (5.2) can be decomposed as

$$\begin{aligned}\mathrm{P}(Y(t))F(t, Y(t)) = \sum_{i=1}^{d-1} \mathrm{Ten}_i \Big[\Big(\mathbf{I}_{n_i} \otimes \mathbf{U}_{i-1} \, \mathbf{U}_{i-1}^\top \Big) \, \mathbf{Unf}_i\big(F(t, Y(t))\big) \, \mathbf{V}_{i+1} \, \mathbf{V}_{i+1}^\top \\ - \mathbf{U}_i \, \mathbf{U}_i^\top \, \mathbf{Unf}_i\big(F(t, Y(t))\big) \, \mathbf{V}_{i+1} \, \mathbf{V}_{i+1}^\top \Big] \\ + \mathrm{Ten}_d \Big[\Big(\mathbf{I}_{n_d} \otimes \mathbf{U}_{d-1} \, \mathbf{U}_{d-1}^\top \Big) \, \mathbf{Unf}_d\big(F(t, Y(t))\big) \Big] ,\end{aligned}$$

see [LOV15, Theorem 3.1]. The orthogonal projections $\mathrm{P}_{\mathbf{U}_i} = \mathbf{U}_i \, \mathbf{U}_i^\top$ and $\mathrm{P}_{\mathbf{V}_i} = \mathbf{V}_i \, \mathbf{V}_i^\top$ operate on the core tensors C_k with $k = 1, \ldots, i$ and on the core tensors C_l with $l = i, \ldots, d$, respectively. Denoting

$$\begin{aligned} & \mathrm{P}_i^+(Y_i^+(t))F(t, Y_i^+(t)) = \mathrm{Ten}_i \left[\big(\mathbf{I}_{n_i} \otimes \mathrm{P}_{\mathbf{U}_{i-1}} \big) \, \mathbf{Unf}_i\big(F(t, Y_i^+(t))\big) \mathrm{P}_{\mathbf{V}_{i+1}} \right] \\ \text{and} \quad & \mathrm{P}_i^-(Y_i^-(t))F(t, Y_i^-(t)) = \mathrm{Ten}_i \left[\mathrm{P}_{\mathbf{U}_i} \, \mathbf{Unf}_i\big(F(t, Y_i^-(t))\big) \mathrm{P}_{\mathbf{V}_{i+1}} \right] , \end{aligned}$$

where for notational simplicity we left out the time dependence of the orthogonal projections $\mathrm{P}_{\mathbf{U}_i}$ and $\mathrm{P}_{\mathbf{V}_i}$, respectively, we rewrite the differential equation for $Y(t)$ as

$$\dot{Y}(t) = \mathrm{P}_1^+(Y_1^+(t))F(t, Y_1^+(t)) - \mathrm{P}_1^-(Y_1^-(t))F(t, Y_1^-(t)) + \cdots + \mathrm{P}_d^+(Y_d^+(t))F(t, Y_d^+(t)). \tag{5.3}$$

This decomposed right-hand side of the differential equation is similar to the form (2.6) in the matrix case. In fact, for $d = 2$, this is the actual decomposed differential equation for the matrix $\mathbf{Y}(t)$.

We do not solve this evolution equation directly, but we follow the Lie–Trotter splitting method, where we solve the $(2d-1)$ subproblems

$$\begin{aligned}
&\dot{Y}_1^+(t) = \mathrm{P}_1^+(Y_1^+(t))F(t,Y_1^+(t)), && Y_1^+(t_0) = Y_0, \\
&\dot{Y}_i^-(t) = -\mathrm{P}_i^-(Y_i^-(t))F(t,Y_i^-(t)), && Y_i^-(t_0) = Y_i^+(t_0+h) && \text{for } i = 1,\dots,d-1, \\
&\dot{Y}_i^+(t) = \mathrm{P}_i^+(Y_i^+(t))F(t,Y_i^+(t)), && Y_i^+(t_0) = Y_{i-1}^-(t_0+h) && \text{for } i = 2,\dots,d
\end{aligned}$$

sequentially. An efficient implementation and a graphical description of the integrator using tensor networks are given in [LOV15].

Now, let us exemplify an important observation about the projections within those subproblems by considering the solutions of the i^--subproblems that are given in [LOV15, Theorem 4.1]. Starting from

$$Y_i^-(t_0) = \mathrm{Ten}_i\left[\mathbf{U}_i(t_0)\mathbf{S}_i(t_0)\,\mathbf{V}_{i+1}^\top(t_0)\right],$$

the solution after one time step is given by

$$\begin{aligned}
Y_i^-(t_1) &= Y_i^-(t_0) - \int_{t_0}^{t_1} \mathrm{P}_i^-(Y_i^-(t_0))F(t,Y_i^-(t))\,\mathrm{d}t \\
&= \mathrm{Ten}_i\left[\mathbf{U}_i(t_0)\mathbf{S}_i(t_0)\,\mathbf{V}_{i+1}^\top(t_0) - \int_{t_0}^{t_1} \mathbf{P}_{\mathbf{U}_i}\,\mathbf{Unf}_i\left[F(t,Y_i^-(t))\right]\mathbf{P}_{\mathbf{V}_{i+1}}\,\mathrm{d}t\right] \\
&= \mathrm{Ten}_i\left[\mathbf{U}_i(t_0)\left(\mathbf{S}_i(t_0) - \int_{t_0}^{t_1} \mathbf{U}_i^\top(t_0)\,\mathbf{Unf}_i\left[F(t,Y_i^-(t))\right]\mathbf{V}_{i+1}(t_0)\,\mathrm{d}t\right)\mathbf{V}_{i+1}^\top(t_0)\right].
\end{aligned}$$

Comparing the initial value with the solution, we observe that the factor matrices $\mathbf{U}_i(t_0)$ and $\mathbf{V}_{i+1}(t_0)$ stay constant and that in fact, we have updated the matrix $\mathbf{S}_i(t)$. Since the orthogonal projections consist of those two matrices, we conlude that, as in the matrix case, the projections $\mathrm{P}_i^\pm$ are preserved during the solution of the corresponding subproblem. The same observation is made for the closed-form solution of the i^+-subproblems, see [LOV15, Theorem 4.1].

Another essential property of the tensor train integrator is its exactness, which holds true for matrices and Tucker tensors, see Section 2.2.1, Section 4.4 and Section 4.6.3. It is also valid, under the same assumptions, for tensor trains [LOV15, Theorem 5.1]: if the explicitly given tensor $A(t)$ is in the manifold of low rank TT tensors and $A(t_0) = Y^0$, then, for sufficiently small time steps $h > 0$, the tensor train projector-splitting integrator is exact, i.e., $Y_d^+(t_1) = A(t_1)$ after one time step.

Just as in the matrix case, the preservation of $\mathbf{U}_i$ and $\mathbf{V}_i$ in the integration steps for all $i = 1,\dots,d$ as well as the exactness property are the two essential ingredients to prove robustness also for the tensor train projector-splitting integrator with respect to small singular values.

Assumption 5.1. *We assume that*

(1) F is Lipschitz continuous:

$$\|F(t,Y) - F(t,\widetilde{Y})\| \le L\|Y - \widetilde{Y}\| \qquad \forall\, Y,\widetilde{Y} \in \mathbb{R}^{n_1\times\cdots\times n_d},$$

(2) F is bounded:

$$\|F(t,Y)\| \leq B \qquad \forall\, Y \in \mathbb{R}^{n_1 \times \cdots \times n_d},$$

(3) F is in the tangent space $\mathcal{T}_Y\mathcal{M}$ up to a small perturbation term:

$$F(t,Y) = M(t,Y) + R(t,Y),$$

where M maps to the tangent bundle of the low-rank manifold $\mathcal{M}$ and the remainder R is small on $\mathcal{M}$,

$$M(t,Y) \in \mathcal{T}_Y\mathcal{M} \quad \text{and} \quad \|R(t,Y)\| \leq \varepsilon \quad \forall\, Y \in \mathcal{M}, \quad \forall\, t_0 \leq t \leq T,$$

(4) the initial value $A^0 \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ and the starting value $Y^0 \in \mathcal{M}$ of the numerical method are δ-close:

$$\|A^0 - Y^0\| \leq \delta.$$

With these assumptions at hand, we are now in the position to state the error bound drawn from [KLW16, Theorem 3.1] of the dynamical low-rank approximation of tensor trains when applying the projector-splitting integrator.

Theorem 5.2. *Under Assumption 5.1, the error of the Lie–Trotter and of the Strang splitting method at $t_n = t_0 + nh$, with step size $h > 0$, is bounded by*

$$\|A(t_n) - Y^n\| \leq c_0\delta + c_1\varepsilon + c_2 h \qquad \text{for } t_n \leq T,$$

where the constants c_i only depend on L, B, T and the dimension d.

A proof of this error estimate is given in [KLW16]. We give a concise overview of the strategy of the proof.

In order to solve the differential equation (5.3) for $Y(t)$, we apply the d-dimensional projector-splitting integrator and solve the subproblems for $Y_i^{\pm}(t)$ one after the other. Starting with $i = 1$, the trick is to compress the remaining projections for $i = 2, \ldots, d$ and identify it as one projection in merged form, such that we have to solve three subpoblems, just as in the matrix case. The first two steps of this integrator are the same as in the matrix case, they yield $Y_1^+(t_1)$ and $Y_1^-(t_1)$, respectively. Now, since the arising third subproblem with the compressed projection is prohibitively large, we do not solve it directly, but we perform a low-rank approximation by applying the $(d-1)$-dimensional projector-splitting integrator for updating the remaining core tensors. Due to the exactness result and the preservation of the subprojections mentioned above, the substeps of the $(d-1)$-dimensional tensor-train projector-splitting integrator are the same as those from the full d-dimensional integrator but starting from the third substep onwards. This observation is the key to the main idea of recursively using the error bound of the matrix case with inexact solution of the third step, see Section 2.4.2. Since we solve the third step by a low-rank approximation, we obtain an error of size $\mathcal{O}(\delta+\varepsilon+h+\eta)$, where η represents the additional error obtained

from the approximate solution of the third step. This third step consists of applying the $(d-1)$-dimensional TT projector-splitting integrator and so in fact the application of the matrix integrator for updating the core in the next mode. Therefore, η is of the form $\eta = \mathcal{O}(\varepsilon + h)$. Continuing this argument until the last step $i = d$, an error bound of size $\mathcal{O}(\delta + \varepsilon + h)$ follows.

The numerical example in [KLW16, Section 4.2] corroborates this convergence result.

5.2 Outlook: Time integration of tensor tree networks

Tensor trains can be interpreted as a special case to the recently proposed hierarchical Tucker (HT) tensors, independently presented in [HK09] and [Gra10]. For a subspace based treatment of representations of higher-order tensors, we refer to [BSU16]. Hierarchical Tucker tensors employ a recursive hierarchical construction of Tucker tensor type. Their decomposition is introduced based on a binary dimension tree, which we exemplify for a three-dimensional case in the following figure:

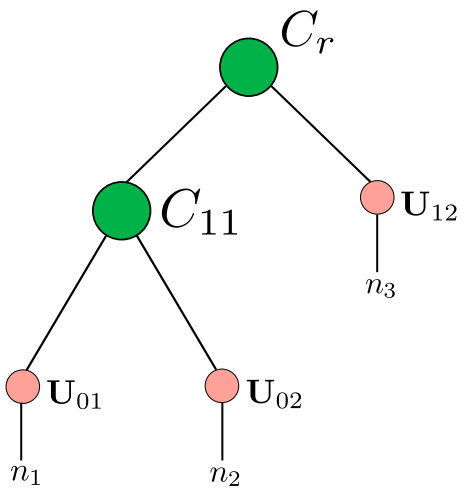

Figure 5.1: Exemplary three-dimensional binary tree for HT.

Analogous to the Tucker format, the manifold of hierarchical Tucker tensors can be shown to be an embedded manifold [UV13]. Applying the dynamical low-rank approximation ansatz to the tensor differential equation (5.1) with the underlying HT manifold again yields a tensor differential equation for the approximate HT tensor $Y(t)$. A numerical integration procedure for integrating HT tensors in time is proposed in [LRSV13]. Similarly as in the numerical analysis of the integration method in [KL07], this integrator for HT tensors is shown to have curvature bounds and error estimates which depend on the inverse of the smallest singular value of matricizations of the hierarchical Tucker tensor, see [LRSV13, AJ14].

It would thus be interesting to extend the HT integrator in [LRSV13] to a method that is robust with respect to small singular values.

Even more general than the HT tensors are tensor tree networks (TTN), whose dimen-

sion tree is not based on two, but on several branches at each node. We exemplify this by extending the binary tree for HT tensors in Figure 5.1 to a general tree for TTN in the following figure:

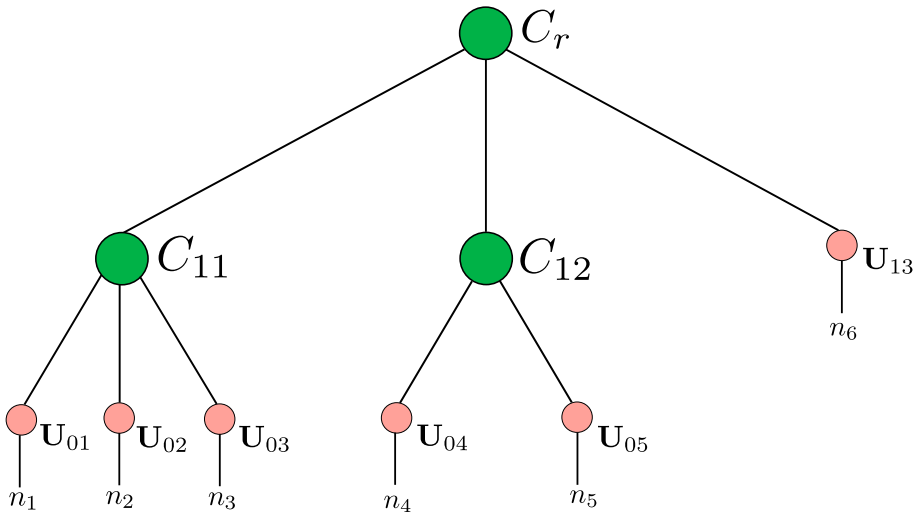

Figure 5.2: Exemplary six-dimensional general tree for TTN.

In this thesis, we have proposed the nested Tucker integrator for integrating Tucker tensors in time. It is based on the idea of recursively solving the first two steps of the matrix projector-splitting integrator directly and the third step by performing a low-rank approximation for each mode $i = 1, \ldots, d$. Now, rewriting tensor tree networks in a beneficial way as Tucker tensors allows us to transfer the idea of applying the matrix projector-splitting integrator with inexact solutions of substeps to the TTN case. Since this low-rank tensor format is a generalization of many underlying tensor formats, such as Tucker tensors or tensor trains, and due to its promising reduction of computational cost, its development is of large interest. A robust and efficient time integrator for TTN is a topic of current research that will be reported elsewhere.

Bibliography

[AG15] P. Axelsson and F. Gustafsson: Discrete-time solutions to the continuous-time differential Lyapunov equation with applications to Kalman filtering. *IEEE Transactions on Automatic Control*, 60:632–643, 2015.

[AJ14] A. Arnold and T. Jahnke: On the approximation of high-dimensional differential equations in the hierarchical Tucker format. *BIT Numerical Mathematics*, 54:305–341, 2014.

[Ale61] V.M. Alekseev: An estimate for the perturbations of the solutions of ordinary differential equations. *Westnik Moskov Univ. Ser. I Mat. Meh. 2*, 1:28–36, 1961.

[AMH11] A.H. Al-Mohy and N.J. Higham: Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM Journal on Scientific Computing*, 33:488–511, 2011.

[AO15] P.-A. Absil and I.V. Oseledets: Low-rank retractions: a survey and new results. *Computational Optimization and Applications*, 62:5–29, 2015.

[Arn51] W.E. Arnoldi: The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9:17–29, 1951.

[Baş91] T. Başar: Generalized Riccati equations in dynamic games. In: S. Bittanti, A.J. Laub and J.C. Willems (editors), *The Riccati Equation*. Springer, Berlin Heidelberg, 293–333, 1991.

[Bel73] E. Beltrami: Sulle funzioni bilineari. *Giornale di Matematiche ad Uso degli Studenti Delle Università*, 11:98–106, 1873.

[Bel61] R.E. Bellman: *Adaptive control processes: a guided tour*. Princeton University Press, Princeton, NJ, 1961.

[BJWM00] M.H. Beck, A. Jäckle, G.A. Worth and H.-D. Meyer: The multiconfiguration time-dependent Hartree (MCTDH) method: a highly efficient algorithm for propagating wavepackets. *Elsevier Physics reports*, 324:1–105, 2000.

[BL18] P. Benner and N. Lang: Peer methods for the solution of large-scale differential matrix equations. *arXiv preprint* `arXiv:1807.08524v1 [math.NA]`, 2018.

[BM97] M.H. Beck and H.-D. Meyer: An efficient and robust integration scheme for the equations of motion of the multiconfiguration time-dependent Hartree (MCTDH) method. *Zeitschrift für Physik D Atoms, Molecules and Clusters*, 42:113–129, 1997.

[BM17] T. Başar and J. Moon: Riccati equations in Nash and Stackelberg differential and dynamic games. In: D. Dochain, D. Henrion and D. Peaucelle (editors), *IFAC-PapersOnLine*. Elsevier, Amsterdam, 2017, 9547–9554.

[Boy01] J.P. Boyd: *Chebyshev and Fourier spectral methods*. Dover, Mineola, NY, 2nd edition, 2001.

[BSU16] M. Bachmayr, R. Schneider and A. Uschmajew: Tensor networks and hierarchical tensors for the solution of high-dimensional partial differential equations. *Foundations of Computational Mathematics*, 16:1423–1472, 2016.

[CC70] J.D. Carroll and J.-J. Chang: Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35:283–319, 1970.

[CC08] K. Choi and A. Cichocki: Control of a wheelchair by motor imagery in real time. In: C. Fyfe, D. Kim, S.-Y. Lee and H. Yin (editors), *Intelligent Data Engineering and Automated Learning – IDEAL 2008. IDEAL 2008*, Lecture Notes in Computer Science vol. 5326. Springer, Berlin Heidelberg, 2008, 330–337.

[Cic13] A. Cichocki: Tensor decompositions: a new concept in brain data analysis?. *arXiv preprint* `arXiv:1305.0395v1 [cs.NA]`, 2013.

[CKOR16] M. Caliari, P. Kandolf, A. Ostermann and S. Rainer: The Leja method revisited: backward error analysis for the matrix exponential. *SIAM Journal on Scientific Computing*, 38:A1639–A1661, 2016.

[CLK+15] F. Cong, Q.-H. Lin, L.-D. Kuang, X.-F. Gong, P. Astikainen and T. Ristaniemi: Tensor decomposition of EEG signals: a brief review. *Journal of Neuroscience Methods*, 248:59–69, 2015.

[CWR+08] A. Cichocki, Y. Washizawa, T. Rutkowski, H. Bakardjian, A.-H. Phan, S. Choi, H. Lee, Q. Zhao, L. Zhang and Y. Li: Noninvasive BCIs: Multiway signal-processing array decompositions. *Computer*, 41:34–42, 2008.

[CZPA09] A. Cichocki, R. Zdunek, A.-H. Phan and S.-i. Amari: *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation.* John Wiley & Sons, West Sussex, 2009.

[DE99] L. Dieci and T. Eirola: On smooth decompositions of matrices. *SIAM Journal on Matrix Analysis and Applications*, 20:800–819, 1999.

[Dir30a] P.A.M. Dirac: Note on exchange phenomena in the Thomas atom. *Mathematical Proceedings of the Cambridge Philosophical Society*, 26:376–385, 1930.

[Dir30b] P.A.M. Dirac: *The principles of quantum mechanics.* Clarendon press, Oxford, 1930.

[DK90] J. Demmel and W. Kahan: Accurate singular values of bidiagonal matrices. *SIAM Journal on Scientific and Statistical Computing*, 11:873–912, 1990.

[DLDMV00a] L. De Lathauwer, B. De Moor and J. Vandewalle: A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21:1253–1278, 2000.

[DLDMV00b] L. De Lathauwer, B. De Moor and J. Vandewalle: On the best rank-1 and rank-$(\mathrm{R}_1, \mathrm{R}_2, ..., \mathrm{R}_n)$ approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21:1324–1342, 2000.

[DS87] C.E. De Souza: Riccati differential equation in optimal filtering of periodic non-stabilizable systems. *International Journal of Control*, 46:1235–1250, 1987.

[EN99] K.-J. Engel and R. Nagel: *One-parameter semigroups for linear evolution equations.* Springer Science & Business Media, New York, 1999.

[EN06] K.-J. Engel and R. Nagel: *A short course on operator semigroups.* Springer Science & Business Media, New York, 2006.

[EO13] L. Einkemmer and A. Ostermann: Exponential integrators on graphic processing units. In: *High performance computing and simulation (HPCS)*, 2013 International Conference on High Performance Computing & Simulation. IEEE, New York, 490–496, 2013.

[EO15] L. Einkemmer and A. Ostermann: Overcoming order reduction in diffusion-reaction splitting. Part 1: Dirichlet boundary conditions. *SIAM Journal on Scientific Computing*, 37:A1577–A1592, 2015.

[EY36] C. Eckart and G. Young: The approximation of one matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936.

[Fre34] J.I. Frenkel: *Wave mechanics: advanced general theory.* Clarendon Press, Oxford, 1934.

[Gau09] C.F. Gauss: *Theoria motus corporum coelestium in sectionibus conicis solem ambientium.* Perthes and Besser, Hamburg, 1809.

[Gau23] C.F. Gauss: *Theoria combinationis observationum erroribus minimis obnoxiae, pars prior.* Königliche Gesellschaft der Wissenschaften zu Göttingen, Göttingen, 1823.

[GK65] G. Golub and W. Kahan: Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2:205–224, 1965.

[GKT13] L. Grasedyck, D. Kressner and Ch. Tobler: A literature survey of low rank tensor approximation techniques. *GAMM-Mitteilungen*, 36:53–78, 2013.

[Gra10] L. Grasedyck: Hierarchical singular value decomposition of tensors. *SIAM Journal on Matrix Analysis and Applications*, 31:2029–2054, 2010.

[Grö67] W. Gröbner: *Die Lie-Reihen und ihre Anwendungen.* Deutscher Verlag der Wissenschaften, Berlin, 1967.

[GVL96] G.H. Golub and C.F. Van Loan: *Matrix computations.* The Johns Hopkins University Press, Baltimore, MD, 1996.

[Hac12] W. Hackbusch: *Tensor spaces and numerical tensor calculus.* Springer, Berlin Heidelberg, 2012.

[Har70] R.A. Harshman: Foundations of the PARAFAC procedure: models and conditions for an "explanatory" multimodal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.

[Hit27] F.L. Hitchcock: The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6:164–189, 1927.

[Hit28] F.L. Hitchcock: Multiple invariants and generalized rank of a p-way matrix or tensor. *Journal of Mathematics and Physics*, 7:39–79, 1928.

[HK09] W. Hackbusch and S. Kühn: A new scheme for the tensor representation. *Journal of Fourier Analysis and Applications*, 15:706–722, 2009.

[HL97] M. Hochbruck and Ch. Lubich: On Krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 34:1911–1925, 1997.

[HLO+16] J. Haegeman, Ch. Lubich, I.V. Oseledets, B. Vandereycken and F. Verstraete: Unifying time evolution and optimization with matrix product states. *Physical Review B*, 94:165116, 2016.

[HLW06] E. Hairer, Ch. Lubich and G. Wanner: *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*. Springer, Berlin Heidelberg, 2006.

[HM12] U. Helmke and J.B. Moore: *Optimization and dynamical systems*. Springer, London, 2012.

[HNW93] E. Hairer, S.P. Nørsett and G. Wanner: *Solving ordinary differential equations I. Nonstiff problems*. Springer, Berlin Heidelberg, 1993.

[HO10] M. Hochbruck and A. Ostermann: Exponential integrators. *Acta Numerica*, 19:209–286, 2010.

[HOV13] J. Haegeman, T.J. Osborne and F. Verstraete: Post-matrix product state methods: to tangent space and beyond. *Physical Review B*, 88:075133, 2013.

[HRS12] S. Holtz, T. Rohwedder and R. Schneider: On manifolds of tensors of fixed TT-rank. *Numerische Mathematik*, 120:701–731, 2012.

[HS14] E. Hansen and T. Stillfjord: Convergence analysis for splitting of the abstract differential Riccati equation. *SIAM Journal on Numerical Analysis*, 52:3128–3139, 2014.

[JL00] T. Jahnke and Ch. Lubich: Error bounds for exponential operator splittings. *BIT Numerical Mathematics*, 40:735–744, 2000.

[Jor74] C. Jordan: Mémoire sur les formes bilinéaires. *Journal de mathématiques pures et appliquées*, 19:35–54, 1874.

[Kal60] R.E. Kalman: A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

[KB61] R.E. Kalman and R.S. Bucy: New results in linear filtering and prediction theory. *Journal of Basic Engineering*, 83:95–108, 1961.

[KB09] T.G. Kolda and B.W. Bader: Tensor decompositions and applications. *SIAM review*, 51:455–500, 2009.

[KBL17] B. Kloss, I. Burghardt and Ch. Lubich: Implementation of a novel projector-splitting integrator for the multi-configurational time-dependent Hartree approach. *Journal of Chemical Physics*, 146:174107, 2017.

[KK18] V. Khoromskaia and B.N. Khoromskij: *Tensor numerical methods in quantum chemistry*. Walter de Gruyter GmbH & Co KG, Berlin, 2018.

[KL07] O. Koch and Ch. Lubich: Dynamical low-rank approximation. *SIAM Journal on Matrix Analysis and Applications*, 29:434–454, 2007.

[KL10] O. Koch and Ch. Lubich: Dynamical tensor approximation. *SIAM Journal on Matrix Analysis and Applications*, 31:2360–2375, 2010.

[KLW16] E. Kieri, Ch. Lubich and H. Walach: Discretized dynamical low-rank approximation in the presence of small singular values. *SIAM Journal on Numerical Analysis*, 54:1020–1038, 2016.

[Kön03] K. Königsberger: *Analysis 2.* Springer, Berlin Heidelberg, 2003.

[LO14] Ch. Lubich and I.V. Oseledets: A projector-splitting integrator for dynamical low-rank approximation. *BIT Numerical Mathematics*, 54:171–188, 2014.

[LOV15] Ch. Lubich, I.V. Oseledets and B. Vandereycken: Time integration of tensor trains. *SIAM Journal on Numerical Analysis*, 53:917–941, 2015.

[LRSV13] Ch. Lubich, T. Rohwedder, R. Schneider and B. Vandereycken: Dynamical approximation by hierarchical Tucker and tensor-train tensors. *SIAM Journal on Matrix Analysis and Applications*, 34:470–494, 2013.

[LSS16] N. Lang, J. Saak and T. Stykel: Balanced truncation model reduction for linear time-varying systems. *Mathematical and Computer Modelling of Dynamical Systems*, 22:267–281, 2016.

[Lub04] Ch. Lubich: A variational splitting integrator for quantum molecular dynamics. *Applied Numerical Mathematics*, 48:355–368, 2004.

[Lub05] Ch. Lubich: On variational approximations in quantum molecular dynamics. *Mathematics of Computation*, 74:765–779, 2005.

[Lub08] Ch. Lubich: *From quantum to classical molecular dynamics: reduced models and numerical analysis.* European Mathematical Society, Zürich, 2008.

[Lub14] Ch. Lubich: Low-rank dynamics. In: S. Dahlke, W. Dahmen, M. Griebel, W. Hackbusch, K. Ritter, R. Schneider, Ch. Schwab and H. Yserentant (editors), *Extraction of Quantifiable Information from Complex Systems.* Springer, Switzerland, 381–396, 2014.

[Lub15] Ch. Lubich: Time integration in the multiconfiguration time-dependent Hartree method of molecular quantum dynamics. *Applied Mathematics Research eXpress*, 2015:311–328, 2015.

[LVW18] Ch. Lubich, B. Vandereycken and H. Walach: Time integration of rank-constrained Tucker tensors. *SIAM Journal on Numerical Analysis*, 56:1273–1290, 2018.

[Men07] H. Mena: *Numerical solution of differential riccati equations arising in optimal control problems for parabolic partial differential equations.* PhD thesis, Escuela Politecnica Nacional, Quito, 2007.

[Mir60] L. Mirsky: Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics*, 11:50–59, 1960.

[MQ02] R.I. McLachlan and G.R.W. Quispel: Splitting methods. *Acta Numerica*, 11:341–434, 2002.

[NL08] A. Nonnenmacher and Ch. Lubich: Dynamical low-rank approximation: applications and numerical experiments. *Mathematics and Computers in Simulation*, 79:1346–1357, 2008.

[OPW18] A. Ostermann, C. Piazzola and H. Walach: Convergence of a low-rank Lie–Trotter splitting for stiff matrix differential equations. *Preprint* 2018. Available at `https://na.uni-tuebingen.de/~walach/`.

[Ose11] I.V. Oseledets: Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33:2295–2317, 2011.

[OT09] I.V. Oseledets and E.E. Tyrtyshnikov: Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM Journal on Scientific Computing*, 31:3744–3759, 2009.

[Paz83] A. Pazy: *Semigroups of Linear Operators and Applications to Partial Differential Operators*. Springer, New York, 1983.

[PGVWC06] D. Perez-Garcia, F. Verstraete, M.M. Wolf and J.I. Cirac: Matrix product state representations. *arXiv preprint* `arXiv:quant-ph/0608197v2`, 2006.

[Rei72] W.T. Reid: *Riccati differential equations*. Academic Press, New York, 1972.

[Saa92] Y. Saad: Analysis of some Krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*: 29:209–228, 1992.

[San04] H. Sandberg: *Model reduction for linear time-varying systems*, PhD thesis, Lund University, Lund, 2004.

[Sch07] E. Schmidt: Zur Theorie der linearen und nichtlinearen Integralgleichungen. I. Teil: Entwicklung willkürlicher Funktionen nach Systemen vorgeschriebener. *Mathematische Annalen*, 63:433–476, 1907.

[Sch26] E. Schrödinger: An undulatory theory of the mechanics of atoms and molecules. *Physical review*, 28:1049–1070, 1926.

[Sch11] U. Schollwöck: The density-matrix renormalization group in the age of matrix product states. *Annals of Physics*, 326:96–192, 2011.

[Ste93] G.W. Stewart: On the early history of the singular value decomposition. *SIAM review*, 35:551–566, 1993.

[Sti35] E. Stiefel: Richtungsfelder und Fernparallelismus in n-dimensionalen Mannigfaltigkeiten. *Commentarii Mathematici Helvetici*, 8:305–353, 1935.

[Sti15] T. Stillfjord: Low-rank second-order splitting of large-scale differential Riccati equations. *IEEE Transactions on Automatic Control*, 60:2791–2796, 2015.

[Str68] G. Strang: On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5:506–517, 1968.

[Tay17] B. Taylor: *Methodus incrementorum directa & inversa*. Inny, London, 1717.

[TB97] L.N. Trefethen and D. Bau III: *Numerical linear algebra*. SIAM, Philadelphia, PA, 1997.

[TS01] A. Trombettoni and A. Smerzi: Discrete solitons and breathers with dilute Bose–Einstein condensates. *Physical Review Letters*, 86:2353–2356, 2001.

[Tuc66] L.R. Tucker: Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966.

[UV13] A. Uschmajew and B. Vandereycken: The geometry of algorithms using hierarchical tensors. *Linear Algebra and its Applications*, 439:133–166, 2013.

[VMC08] F. Verstraete, V. Murg and J.I. Cirac: Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems. *Advances in Physics*, 57:143–224, 2008.

[VVM12] N. Vannieuwenhoven, R. Vandebril and K. Meerbergen: A new truncation strategy for the higher-order singular value decomposition. *SIAM Journal on Scientific Computing*, 34:A1027–A1052, 2012.