Optimization in Tensor Spaces for Data Science and Scientific Computing

dissertation

zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften

vorgelegt am Fachbereich IV
der Universität Trier

von

Gennadij Heidel

Trier, im Januar 2019

Abstract

In this thesis, we consider the solution of high-dimensional optimization problems with an underlying low-rank tensor structure. Due to the exponentially increasing computational complexity in the number of dimensions—the so-called *curse of dimensionality*—they present a considerable computational challenge and become infeasible even for moderate problem sizes.

Multilinear algebra and tensor numerical methods have a wide range of applications in the fields of data science and scientific computing. Due to the typically large problem sizes in practical settings, efficient methods, which exploit low-rank structures, are essential. In this thesis, we consider an application each in both of these fields.

Tensor completion, or imputation of unknown values in partially known multiway data is an important problem, which appears in statistics, mathematical imaging science and data science. Under the assumption of redundancy in the underlying data, this is a well-defined problem and methods of mathematical optimization can be applied to it. Due to the fact that tensors of fixed rank form a Riemannian submanifold of the ambient high-dimensional tensor space, Riemannian optimization is a natural framework for these problems, which is both mathematically rigorous and computationally efficient. We present a novel Riemannian trust-region scheme, which compares favourably with the state of the art on selected application cases and outperforms known methods on some test problems.

Optimization problems governed by partial differential equations form an area of scientific computing which has applications in a variety of areas, ranging from physics to financial mathematics. Due to the inherent high dimensionality of optimization problems arising from discretized differential equations, these problems present computational challenges, especially in the case of three or more dimensions. An even more challenging class of optimization problems has operators of integral instead of differential type in the constraint. These operators are nonlocal, and therefore lead to large, dense discrete systems of equations. We present a novel solution method, based on separation of spatial dimensions and provably low-rank approximation of the nonlocal operator. Our approach allows the solution of multidimensional problems with a complexity which is only slightly larger than linear in the univariate grid size; this improves the state of the art for a particular test problem problem by at least two orders of magnitude.

Zusammenfassung

In dieser Arbeit betrachten wir die Lösung hochdimensionaler Optimierungsprobleme, welchen eine Tensorstruktur niedrigen Ranges zugrunde liegt. Da die rechnerische Komplexität exponentiell mit der Anzahl der Dimensionen anwächst – dies wird als *curse of dimensionality* („Fluch der Dimensionalität“) bezeichnet – stellen sie eine beträchtliche rechnerische Herausforderung dar und hören bereits für moderate Problemgrößen auf handhabbar zu sein.

Multilineare Algebra und numerische Methoden für Tensoren haben eine Reihe von Anwendungen in den Gebieten von Data Science und des wissenschaftlichen Rechnens. Da praktische Probleme typischerweise eine hohe Dimension haben, sind effiziente Methoden, die Niedrigrangstrukturen ausnutzen, von essentieller Bedeutung. In dieser Arbeit betrachten wir jeweils eine Anwendung in diesen beiden Gebieten.

Tensorvervollständigung, oder Imputation unbekannter Werte in teilweise bekannten mehrdimensionalen Datensätzen ist ein wichtiges Problem in der Statistik, der mathematischen Bildverarbeitung und in Data Science. Wenn man Redundanz der zugrundeliegenden Daten annimmt, ist es möglich, Verfahren der mathematischen Optimierung für dieses Problem anzuwenden. Da Tensoren festen Ranges eine Riemann'sche Mannigfaltigkeit des umgebenden hochdimensionalen Tensorraums darstellen, ist die Riemann'sche Optimierung der natürliche Rahmen für die Behandlung dieser Probleme, der sowohl mathematisch rigoros als auch rechnerisch effizient ist. Wir stellen ein neuartiges Riemann'sches Trust-Region-Verfahren vor, welches dem Vergleich mit dem neuesten Stand der Forschung für ausgewählte Anwendungen standhält sowie eine Verbesserung bekannter Methoden für einige Testprobleme bietet.

Optimierungsprobleme mit partiellen Differentialgleichungen als Restriktion stellen ein Gebiet des wissenschaftlichen Rechnens dar, welches Anwendungen in zahlreichen Gebieten hat, von der Physik bis hin zur Finanzmathematik. Da Optimierungsprobleme, die von Differentialgleichungen herrühren, inhärent hohe Dimension haben, stellen diese Probleme eine rechnerische Herausforderung dar, insbesondere im Fall von drei oder mehr Dimensionen. Eine noch schwierigere Klasse von Problemen hat Operatoren vom Integral- statt vom Differentialtyp in der Restriktion. Diese Operatoren sind nichtlokal und führen somit zu großen, vollbesetzten Gleichungssystemen. Wir präsentieren ein neuartiges Lösungsverfahren, welches auf der Trennung der Raumvariablen und auf einer Approximation für den lokalen Operator, welche beweisbar niedrigen Rang hat, basiert. Unser Ansatz erlaubt es, mehrdimnsionale Probleme mit einer Komplexität zu lösen, die nur wenig höher als linear in der eindimensionale Gitterweite ist; dies verbessert den Stand der Forschung für ein bestimmtes Testproblem um mindestens zwei Größenordnungen.

"Young man, in mathematics you don't understand things. You just get used to them."

– John von Neumann

Acknowledgments

First of all, I would like to express my deepest gratitude to my supervisor Prof. Dr. Volker Schulz for many helpful discussions and his exceptional support throughout my PhD studies. Not only the pleasant working environment in our research group but also the opportunity and his encouragement to present my work at several international conferences and to visit leading researchers have always helped me greatly in my work. I also thank Dr. Jan Pablo Burgard for agreeing to co-supervise my PhD project and for the insightful discussions and his help in finding suitable test data for my work.

I am also very grateful to Dr. Venera Khoromskaia and DrSci. Boris Khoromskij for their hospitality during my visits at the Max Planck Institute for Mathematics in the Sciences in Leipzig. Our collaboration has been most inspiring from both a personal and a professional perspective. A special thanks goes to Boris for agreeing to make the trip to Trier and serve as a referee for my thesis.

I have profited from discussions with Prof. Wen Huang and Prof. Bart Vandereycken during my visit at the University of Geneva; this collaboration has helped me greatly improve my understanding of Riemannian optimization.

I owe thanks to the DFG Research Training Group 2126 'Algorithmic Optimization' (ALOP), which employed me as a research assistant at Trier University during my PhD studies. I also thank all members of the research training group for many fruitful discussions.

Moreover, I thank all my colleagues from the department of mathematics at Trier University; I could not have wished for a more pleasant working environment. Especially, I would like to thank my officemates Dr. Christina Schenk and Christian Vollmann for the nice working atmosphere we had and some serious and many not-so-serious discussions over coffee and gummy bears.

I am grateful to my brother Andrej, who has been with me from the very beginning of my studies and has always served as motivation and both emotional and academic support. I wish to express my special thanks to my wife Anna for always believing in me—even when I did not—and for her extraordinary patience and forbearance, especially during the final months of my PhD time. Thank you for our ongoing journey together.

Zuletzt möchte ich meinen Eltern für ihre Unterstützung danken – immer und in allen Lebenslagen.

Preface

As part of this thesis, a refereed publication was written, which appears as [38] in the bibliography. Furthermore, a preprint [37] has been submitted to a journal. These works are fundamental for the thesis and therefore integrated; this includes figures and results of numerical tests.

Contents

List of Figures

List of Tables

1 Introduction

High-dimensional problems frequently appear in scientific applications. More than six decades ago, Bellman [7] has coined the phrase of the "curse of dimensionality"—the exponential growth of the problem size with respect to dimension.

This "curse" is far worse than the typically polynomial growth of the problem size with respect to the degrees of freedom *within* one dimension and makes even seemingly small problems infeasible if the number of dimensions is large enough. As a simple example, take a problem with 2 degrees of freedom in each dimension and 100 dimensions in total. This leads to a total number of

$$2^{100} \approx 1.27 \times 10^{30}$$

degrees of freedom. The currently (November 2018) fastest supercomputer in the world, the *Summit* at Oak Ridge National Laboratory, has a computing power of 143.5 PetaFLOPS, which means it can perform 143.5×10^{15} floating point operations in one second. Even if all it had to do was to perform one operation on each entry of the tensor introduced above—where we do not even discuss how to store this tensor—the whole operation would take more than $280,000$ years. Therefore, fast and efficient computation is essential when dealing with data of very high dimension. The central idea is to exploit *low-rank* structures, i. e. an inherently redundant structure in the data.

The discipline dealing with multiway data, *multilinear algebra*, goes back many decades. In the most general sense, it deals with the mathematical properties of multiway arrays—*tensors*. For most of its history, multilinear algebra has been driven by the application side; the earliest papers in this field, which laid the foundations for the modern tensor formats come from fields such as chemometrics, psychometrics and phonetics. More recently, the field of (numerical) multilinear algebra has attracted the interest of researchers in mathematics. Of particular importance are the papers [64]—an original research paper from 2000 which put the ideas and heuristics from application sciences on mathematically sound foundations—and [56]—a survey from 2009 which helped popularize the field of multilinear algebra among applied mathematicians. Many recent breakthroughs in the mathematical research on tensors are collected in the book [32].

One of the ideas introduced by mathematicians to the field of multilinear algebra that have led to new advances is *Riemannian optimization*—it can be shown that sets of tensors form a low-dimensional surface in a high-dimensional space, and this property can be exploited to construct especially fast optimization methods. This framework has existed in the matrix case for several years [2], and various generalizations to the tensor case have been proposed in the literature. In this thesis, we will present the first result, which allows to construct exact second-order optimization methods on tensor

manifolds in this framework, and we will show the effectiveness of this method for *tensor completion*—the reconstruction of a partially known low-rank tensor.

Even more recently, connections between the fields of multilinear algebra and scientific computing—the modelling, simulation and optimization with partial differential equations—have gained interest in the mathematical literature. A collection of first results in this field can be found in the two books [47] and [50]. The first mentioned book deals with the computational and applied side of this field, the second one with the theoretical side. Scientific computing presents special challenges to the field of tensor numerical methods, since not only the dimension is high, but also the number of degrees of freedom in each dimension—the typical case is the discretization of PDEs, which easily admits several thousand degrees of freedom in each spatial direction.

A particularly challenging field of scientific computing is given by the so-called *fractional* or *non-local* differential equations. These lead not only to huge systems of equations but also to dense ones, which causes traditional methods to break down even on moderately fine grids. We present a novel method for optimal control of a fractional Laplacian equation which allows performing all computations in just one dimension (with small overhead cost) by approximately separating the spatial variables.

The rest of this thesis is organized as follows. In Chapter 2, we introduce the two best-known tensor formats—the CP and the Tucker format. We also define basic tensor operations and present simple algorithms for computing tensor decompositions and approximations. In Chapter 3, we introduce basic concepts of Riemannian optimization and the framework of optimization on Riemannian manifolds. In Chapter 4, we apply Riemannian optimization to the problem of tensor completion—one of the best known application cases of multilinear algebra in recent years. We show that the second derivative of functions on the given tensor manifold can be computed efficiently to construct a second-order optimization method and present convincing numerical results. In Chapter 5, we consider an optimization problem with a fractional PDE in the constraint and show how tensor methods can be applied to optimal control problems governed by non-local PDEs. Our novel approach allows to solve this three-dimensional problem in one dimension (with some moderate overhead cost) and thus beat existing methods by almost two orders of magnitude. Finally, in Chapter 6, we summarize the two main contributions of this thesis.

2 Low-Rank Tensor Formats

In this chapter, we will present some basic definitions and notions about tensors, especially those of low rank. Throughout this thesis, we will denote tensors

$$\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$$

of *order* $d \geq 3$ by upper-case boldface letters. The entries of the tensor $\mathbf{A}$ are given by $a_{i_1 \cdots i_d}$. As usual, we will denote matrices (which can be seen as tensors of order 2)

$$A \in \mathbb{R}^{m \times n}$$

by upper-case italic letters with entries a_{ij}, and column vectors (which can be seen as tensors of order 1)

$$\mathbf{a} \in \mathbb{R}^{n}$$

by lower-case boldface letters with entries a_i.

In Section 2.1, we will introduce basic tensor formats and discuss their similarities and differences. In Section 2.2, we will present the "workhorse" algorithms for computing these decompositions.

2.1 Basic Notations and Low-Rank Tensor Formats

In this section, we will present the two best-known rank-structured tensor formats, namely the CP decomposition and the Tucker decomposition. We will also introduce some basic notations of tensor algebra we will use throughout the thesis. This section largely follows the exposition in the survey paper by Kolda and Bader [56].

Similarly to matrices, the space $\mathbb{R}^{n_1 \times \cdots \times n_d}$ can be endowed with a *Frobenius inner product*, given by

$$\langle \mathbf{A}, \mathbf{B} \rangle \coloneqq \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} a_{i_1 \cdots i_n} b_{i_1 \cdots i_n}. \tag{2.1}$$

It induces a *Frobenius norm*, given by

$$\|A\| \coloneqq \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle}. \tag{2.2}$$

Since any work with full tensors will be affected by the exponential growth of storage and computing requirements—the curse of dimensionality—the use of low-rank structures is essential. The notion of low rank-tensors is derived from that of low-rank matrices.

In Subsection 2.1.1, we will introduce the simplest low-rank tensors—those of rank one. Starting from here, we will generalize this notion to higher rank in two different ways: in Subsection 2.1.2, we will introduce CP format, and in Subsection 2.1.3 the Tucker format. In Subsection 2.1.4, we will compare the two formats.

2.1.1 Rank-One Matrices and Tensors

Aside from trivial all-zero structures, the simplest low-rank structure in linear algebra is a matrix of rank one. This means that all rows of this matrix are just linear multiples of each other, and so are all the columns. It is also useful to think of such a matrix as an *outer product* of two nonzero vectors, i. e. a matrix $A \in \mathbb{R}^{m \times n}$ is rank one if and only if vectors $\mathbf{u} \in \mathbb{R}^m \setminus \{\mathbf{0}\}$ and $\mathbf{v} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ exist such that

$$A = \mathbf{u}\mathbf{v}^{\mathrm{T}}.$$

The outer product of two vectors can be generalized to any number d of nonzero vectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(d)}$—the result of this operation will then be a tensor of order d. We will denote this as

$$\mathbf{A} = \mathbf{u}^{(1)} \circ \cdots \circ \mathbf{u}^{(d)}, \tag{2.3}$$

and the entries of $\mathbf{A}$ are then given as

$$a_{i_1 \cdots i_d} = u^{(1)}_{i_1} \cdots u^{(d)}_{i_d}.$$

A tensor $\mathbf{A}$ given as in (2.3) is called a *rank-one tensor*.

Clearly, it holds that $\mathbf{u}\mathbf{v}^{\mathrm{T}} = \mathbf{u} \circ \mathbf{v}$, so our definition of rank-one tensors is just an extension of rank-one matrices. Defining tensors of higher rank is more challenging and does not lend itself to such a straightforward generalization of the matrix case. We will discuss this in the following sections.

2.1.2 The Tensor Rank and the CP Decomposition

One possible way to define tensors of rank higher than one is by taking sums of rank-one tensors. A rank-r matrix A admits a decomposition of the form

$$A = UV^{\mathrm{T}}, \tag{2.4}$$

with factor matrices $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{n \times r}$, which both have full column rank. If we consider the individual columns of $U = [\mathbf{u}_1, \ldots, \mathbf{u}_r]$ and $V = [\mathbf{v}_1, \ldots, \mathbf{v}_r]$, (2.4) can also be written as

$$A = \sum_{k=1}^{r} \mathbf{u}_k \mathbf{v}_k^{\mathrm{T}} = \sum_{k=1}^{r} \mathbf{u}_k \circ \mathbf{v}_k, \tag{2.5}$$

so it will be a sum of rank-one "skeletons", see Figure 2.1.

The decomposition (2.5) can be extended in a straightforward fashion to tensors of any order $d \geq 3$. We will consider tensors that admit decompositions of the form

$$\mathbf{A} = \sum_{k=1}^{r} \mathbf{u}_k^{(1)} \circ \cdots \circ \mathbf{u}_k^{(d)}, \tag{2.6}$$

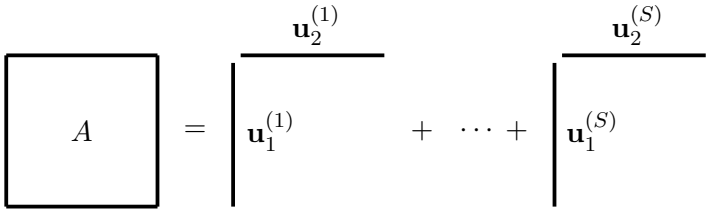


Figure 2.1: Decomposition of a low-rank matrix into a sum of rank-one matrices.

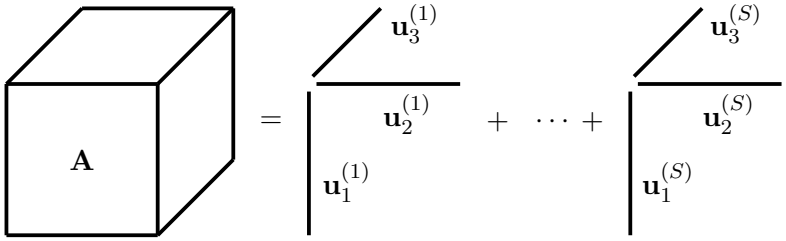


Figure 2.2: CP decomposition of a tensor.

or, if seen elementwise,

$$a_{i_1 \cdots i_d} = \sum_{k=1}^{r} \left(\mathbf{u}_k^{(1)}\right)_{i_1} \cdots \left(\mathbf{u}_k^{(d)}\right)_{i_d}. \tag{2.7}$$

Figure 2.2 illustrates this decomposition. It is clearly seen that the storage complexity is given by $\mathcal{O}(dRn)$, where $n = \max_i n_i$. Any given entry of $\mathbf{A}$ can be computed in dr floating point operations.

The decomposition (2.5) has first been described by Hitchcock in 1927 [41, 42] as the *polyadic form* of a tensor. In the following decades, it has been rediscovered in the literature several times, notably by Carroll and Chang [17] under the name *CANDECOMP* (canonical decomposition), and by Harshman [36] under the name *PARAFAC* (parallel factors), both in 1970. We will also refer to this decomposition as the *CP decomposition*, a term coined by Kiers [53] in 2000—CP can be seen as a combination of CANDECOMP/PARAFAC or as a backronym for *canonical polyadic*.

The CP decomposition motivates the definition of the *tensor rank*, which is one possibility to extend the notion of matrix rank to tensors. Just like in (2.5), the tensor rank of a tensor is the minimal number of rank-one summands needed to express it, i. e.

$$\operatorname{rank}_{\mathrm{T}} \mathbf{A} = \min\{r \mid \mathbf{A} \text{ admits a CP decomposition with } r \text{ summands}\}.$$

To keep the notation concise, we will use a shorthand expression for (2.7): For all modes k, we will collect the factors of the rank-one components into matrices $U_k = [\mathbf{u}_k^{(1)}, \ldots, \mathbf{u}_k^{(d)}]$. Then we will write

$$\mathbf{A} = [\![U_1, \ldots, U_r]\!] \coloneqq \sum_{k=1}^{r} \mathbf{u}_k^{(1)} \circ \cdots \circ \mathbf{u}_k^{(d)}.$$

Occasionally, it will be useful to normalize the columns of the factor matrices U_k to length one and collect the weights into a vector $\boldsymbol{\lambda} \in \mathbb{R}^r$, such that

$$\mathbf{A} = [\![\boldsymbol{\lambda}; U_1, \ldots, U_r]\!] \coloneqq \sum_{k=1}^{r} \lambda_k \, \mathbf{u}_k^{(1)} \circ \cdots \circ \mathbf{u}_k^{(d)}, \ \|\mathbf{u}_k^{(i)}\| = 1, \text{ for all } k, i.$$

While the definition of the tensor rank follows directly from a well-known property of the matrix rank—a rank-r matrix can be decomposed into a sum of r rank-one matrices—the properties of the tensor rank are very different in some respects. A notable example of this is that the tensor rank can be different, depending on the choice of the underlying field—$\mathbb{R}$ or $\mathbb{C}$. Consider, for example, the rank-3 tensor $\mathbf{A} = [\![U_1, U_2, U_3]\!]$, defined by

$$U_1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix}, \ U_2 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \ U_3 = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 1 & 1 \end{bmatrix},$$

see Kruskal [62]. As shown by ten Berge [8], it does indeed not have a CP decomposition with fewer than three summands. However, it can be easily seen the factors

$$V_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -\mathrm{i} & \mathrm{i} \end{bmatrix}, \ V_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ \mathrm{i} & -\mathrm{i} \end{bmatrix}, \ V_3 = \begin{bmatrix} 1 & 1 \\ \mathrm{i} & -\mathrm{i} \end{bmatrix},$$

define a rank-2 decomposition $\mathbf{A} = [\![V_1, V_2, V_3]\!]$ over $\mathbb{C}$.

Another important difference of the tensor rank from the matrix rank concerns the best rank-r approximation problem in the Frobenius norm:

$$\min_{\operatorname{rank}(X) \leq r} \|A - X\|$$

For matrices, it is solved by the *singular value decomposition* (*SVD*): assume that the SVD of a matrix A is given by

$$A = \sum_{k=1}^{s} \sigma_k \, \mathbf{u}_k \circ \mathbf{v}_k,$$

with singular values $\sigma_1 \geq \cdots \geq \sigma_s > 0$. Then the solution for the best rank-r approximation problem (with $r \leq s$) is simply given by

$$A = \sum_{k=1}^{r} \sigma_k \, \mathbf{u}_k \circ \mathbf{v}_k,$$

due to a famous result by Eckart and Young from 1936 [23].

The Eckart–Young result gives an easy framework for switching between best approximations for different ranks: given a best rank-r approximation, we can obtain a best rank-$(r+1)$ approximation by just computing one additional singular value with the corresponding singular vectors. The change from rank r to rank $r-1$ is even trivial by just dropping the last summand. However, this result cannot be generalized to tensors

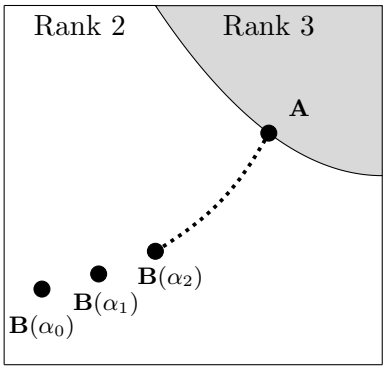


Figure 2.3: Illustration of the tensors $\mathbf{B}(\alpha)$ converging to $\mathbf{A}$ for $\alpha \to \infty$.

of higher order. In [55], Kolda provides an example of a mode-three tensor whose best rank-one approximation is not a summand in the best rank-two approximation.

Aside from these computational difficulties, the best rank-r approximation problem may not even be well-posed, since the set of tensors with bounded rank is, in general, not closed. Therefore, it is practical to call a tensor *degenerate* if it can be approximated to any accuracy by a tensor of lower rank. A well-known example from the literature [76, 80] is given by the rank-three tensor

$$\mathbf{A} = \mathbf{u}_1 \circ \mathbf{v}_1 \circ \mathbf{w}_2 + \mathbf{u}_1 \circ \mathbf{v}_2 \circ \mathbf{w}_1 + \mathbf{u}_2 \circ \mathbf{v}_1 \circ \mathbf{w}_1,$$

where $U \in \mathbb{R}^{n_1 \times 2}$, $V \in \mathbb{R}^{n_2 \times 2}$, $V \in \mathbb{R}^{n_3 \times 2}$ with arbitrarily chosen n_i, and each matrix has linearly independent columns. Now, for $\alpha > 0$, the rank-two tensor

$$\mathbf{B}(\alpha) = \alpha \left(\mathbf{u}_1 + \frac{1}{\alpha}\mathbf{u}_2\right) \circ \left(\mathbf{v}_1 + \frac{1}{\alpha}\mathbf{v}_2\right) \circ \left(\mathbf{w}_1 + \frac{1}{\alpha}\mathbf{w}_2\right) - \alpha\, \mathbf{u}_1 \circ \mathbf{v}_1 \circ \mathbf{w}_1$$

gives us an approximation to $\mathbf{A}$, namely

$$\|\mathbf{A} - \mathbf{B}(\alpha)\| = \frac{1}{\alpha} \left\| \mathbf{u}_2 \circ \mathbf{v}_2 \circ \mathbf{w}_1 + \mathbf{u}_2 \circ \mathbf{v}_1 \circ \mathbf{w}_2 + \mathbf{u}_1 \circ \mathbf{v}_2 \circ \mathbf{w}_2 + \frac{1}{\alpha}\, \mathbf{u}_2 \circ \mathbf{v}_2 \circ \mathbf{w}_2 \right\|,$$

so clearly

$$\lim_{\alpha \to \infty} \|\mathbf{A} - \mathbf{B}(\alpha)\| = 0,$$

although $\operatorname{rank}_{\mathrm{T}} \mathbf{A} = 3 > 2 = \operatorname{rank}_{\mathrm{T}} \mathbf{B}(\alpha)$, for all α.

To make numerical treatment of degenerate tensors feasible, the concept of *border rank* can be introduced, which is given by

$$\begin{aligned} \widetilde{\operatorname{rank}}_{\mathrm{T}}(\mathbf{A}) = \min\{r \,|\, &\text{for any } \varepsilon > 0, \text{ there exists a tensor } \mathbf{E} \\ &\text{such that } \|\mathbf{E}\| < \varepsilon \text{ and } \operatorname{rank}_{\mathrm{T}}(\mathbf{A} + \mathbf{E}) = r\}. \end{aligned}$$

From the definition it follows immediately that

$$\widetilde{\operatorname{rank}}_{\mathrm{T}}(\mathbf{A}) \leq \operatorname{rank}_{\mathrm{T}}(\mathbf{A}).$$

De Silva and Lim [80] also show that the set of degenerate tensors can, in general, have positive volume for at least some ranks; therefore, this is a case which needs to be considered in practice and can cause some difficulties.

2.1.3 The Multilinear Rank and the Tucker Decomposition

As we have seen in the previous section, the CP decomposition poses some numerical problems for the approximation of tensors in low rank. In fact, these problems stretch even further: it can be shown that the problem of determining the tensor rank of a given tensor is NP-complete [44].

In this section, we will present a different generalization of the matrix rank to tensors, which will solve some of the issues with the tensor rank. The SVD of a matrix A can be either written as a sum of rank-one skeleton matrices

$$A = \sum_{k=1}^{r} \sigma_k \, \mathbf{u}_k \circ \mathbf{v}_k,$$

or as a product of the three factor matrices

$$A = U \Sigma V^{\mathrm{T}},$$

where $U \in \mathbb{R}^{m \times r}$ contains the *left singular vectors*, $V \in \mathbb{R}^{n \times r}$ the *right singular vectors* and $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_r)$ the *singular values* of A, see Figure 2.4.

This representation of a low-rank matrix lends itself intuitively to a generalization for tensors, which is illustrated in Figure 2.5 for a mode-three tensor and is called *Tucker decomposition* (introduced in 1966 [86]).

For a formally correct introduction of the Tucker decomposition, we will need some additional notation and definitions. It will be useful to write the entries of a higher-order tensor into a matrix. For a mode-d tensor, there are d natural ways to achieve this: we fix all indices except the kth to obtain a vector—a mode-k *fibre* of a tensor—and then we write all mode-k fibres into a matrix. The result is called the mode-k *matricization* (or *unfolding* or *flattening*) of the tensor.

Formally, the mode-k matricization of a tensor $\mathbf{A}$ is given by

$$A_{(k)} \in \mathbb{R}^{n_k \times \prod_{j \neq k} n_j},$$

such that the row index of $A_{(k)}$ is the kth mode of $\mathbf{A}$, and the column index is a multi-index of the remaining $d-1$ modes. It may be viewed as a d-order generalization of the matrix transpose, since, for $d = 2$, it holds that $A_{(1)} = A$ and $A_{(2)} = A^{\mathrm{T}}$. We denote the re-tensorization of a matricized tensor by a superscript index, i. e. $(A_{(k)})^{(k)} = \mathbf{A}$.

An example helps to illustrate the concept of a matricization. We consider a tensor $\mathbf{A} \in \mathbb{R}^{3 \times 4 \times 2}$, given entrywise by

$$a_{ijk} = i + 3(j-1) + 12(k-1).$$

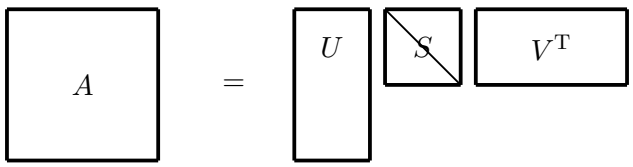


Figure 2.4: SVD of a low-rank matrix.

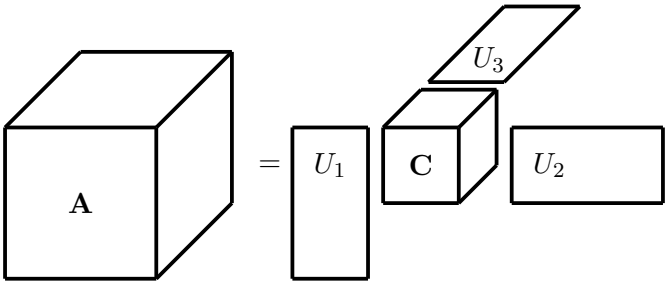


Figure 2.5: Tucker decomposition of a low-rank tensor.

Then the three possible matricizations are

$$A_{(1)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix} \in \mathbb{R}^{3\times 4\cdot 2},$$

$$A_{(2)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{bmatrix} \in \mathbb{R}^{4\times 3\cdot 2}, \text{ and}$$

$$A_{(3)} = \begin{bmatrix} 1 & 2 & 3 & \dots & 10 & 11 & 12 \\ 13 & 14 & 15 & \dots & 22 & 23 & 24 \end{bmatrix} \in \mathbb{R}^{2\times 3\cdot 4}.$$

Now, it is possible to define the *multilinear rank* [62] (or *Tucker rank*) of a tensor as the d-tuple of the ranks of all possible matricizations, i. e.

$$\operatorname{rank}_{\mathrm{ML}}(\mathbf{A}) \coloneqq (\operatorname{rank}(A_{(1)}), \dots, \operatorname{rank}(A_{(d)})).$$

For the matrix case $d = 2$, both the tensor rank and the multilinear rank coincide with the usual notion of the matrix rank because of the well-known fact that the column rank equals the row rank. More precisely, we get

$$\operatorname{rank}_{\mathrm{ML}}(A) = (\operatorname{rank}(A), \operatorname{rank}(A^{\mathrm{T}})) = (\operatorname{rank}(A), \operatorname{rank}(A)) = (\operatorname{rank}_{\mathrm{T}}(A), \operatorname{rank}_{\mathrm{T}}(A)).$$

For tensors of higher order, the different matricization ranks need not coincide. This can already be seen in the very simple example of the tensor $\mathbf{A} \in \mathbb{R}^{2\times 2\times 2}$, given by its mode-1 matricization

$$A_{(1)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

It is readily seen that the other two matricizations are given as

$$A_{(1)} = A_{(2)},\ A_{(3)} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

and therefore the multilinear rank of $\mathbf{A}$ is given as

$$\operatorname{rank}_{\mathrm{ML}}(\mathbf{A}) = (2, 2, 1).$$

To give a formal expression for the Tucker decomposition in Figure 2.5, we will need a notion of a matrix-tensor product. The *i-mode product* of $\mathbf{A}$ with a matrix $M \in \mathbb{R}^{m \times n_i}$ is defined as

$$\mathbf{B} = \mathbf{A} \times_i M \iff B_{(i)} = M A_{(i)}, \ \mathbf{B} \in \mathbb{R}^{n_1 \times \cdots \times n_{i-1} \times m \times n_{i+1} \cdots \times n_d}.$$

It is worth noting that, for different modes, the order of multiplications is irrelevant, i. e.

$$\mathbf{A} \times_i M \times_j N = \mathbf{A} \times_j N \times_i M \quad \text{if } i \neq j. \tag{2.8}$$

If the modes are equal, then

$$\mathbf{A} \times_i M \times_i N = \mathbf{A} \times_i (NM). \tag{2.9}$$

Now, a Tucker decomposition of a tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ with multilinear rank $\mathbf{r} = (r_1, \ldots, r_d)$ is given by

$$\mathbf{X} = \mathbf{C} \times_1 U_1 \cdots \times_d U_d = \mathbf{C} \bigtimes_{i=1}^{d} U_i, \tag{2.10}$$

with a small *core tensor* $\mathbf{C} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ and *basis matrices* $U_k \in \mathbb{R}^{n_k \times r_k}$ with $\operatorname{rank}(U_k) = r_k$. From the definition of the multilinear rank and the k-mode product via matricizations, it is immediately seen that a tensor admits a Tucker decomposition of the form (2.10) if and only if its multilinear rank is equal to $\mathbf{r}$.

Since the core tensor contains $\prod_k r_k$ entries and the basis matrices contain $n_k r_k$ entries each, the overall storage complexity is $\mathcal{O}(r^d + dnr)$, where $n = \max_i n_i$ and $r = \max_i r_i$. A single entry of $\mathbf{A}$ can be computed with the formula

$$a_{i_1 \cdots i_d} = \sum_{k_1=1}^{r_1} \cdots \sum_{k_d=1}^{r_d} c_{k_1 \cdots k_d} u^{(1)}_{i_1 k_1} \cdots u^{(d)}_{i_d k_d} \tag{2.11}$$

in dr^d floating point operations. These expressions show that the Tucker decomposition does not lift the curse of dimensionality, since we still have a complexity which is exponential in d. However, it does ameliorate it, since we have the base r, which we can expect to be much smaller than n in typical applications.

It is worth noting that the Tucker decomposition is not unique. Indeed, for any nonsingular matrices $R_i \in \mathbb{R}^{r_k \times r_k}$, we can get

$$\mathbf{A} = \mathbf{C} \bigtimes_{i=1}^{d} U_i = \Big(\mathbf{C} \bigtimes_{i=1}^{d} R_i\Big) \bigtimes_{i=1}^{d} \big(U_i R_i^{-1}\big). \tag{2.12}$$

In other words, we can modify the core tensor arbitrarily, so long as we apply the inverse modifications to the corresponding basis matrices.

2.1.4 Comparison of Different Low-Rank Formats and Discussion

In the previous two subsections, we have introduced the two oldest and best-known tensor formats. In Subsection 2.1.2, we have seen that the CP format allows a very high compression rate with a simple and intuitive representation, but leads to some mathematical difficulties, such as the possible non-existence of a best low-rank approximation. The Tucker decomposition in Subsection 2.1.3 preserves simplicity and gives us closedness of the bounded-rank set (as we will see later) while preserving simplicity—but it does not lift the curse of dimensionality. In this thesis, we will make use of both concepts when appropriate.

A curiosity of tensor notions for ranks is that the unique definition of rank for matrices leads to different notions in the case $d \geq 2$. Since we have different notions of rank and low-rank decomposition, it makes sense to study the relations between them and the conversion form one format to the other.

First, consider a tensor $\mathbf{A} = [\![U_1, \ldots, U_r]\!]$ with $\operatorname{rank}_{\mathrm{T}}(\mathbf{A}) = r$. Then we can easily transform it into the Tucker format, with the U_is as the basis matrices and the core tensor defined by

$$c_{i_1 \cdots i_d} = \begin{cases} 1, & \text{if } i_1 = \cdots = i_d, \\ 0, & \text{else.} \end{cases} \tag{2.13}$$

Therefore, all matricization ranks of $\mathbf{A}$ are bounded by the tensor rank r. A discussion can be found in [32, Subsection 8.5.2].

Now we consider the other direction. Let $\mathbf{A} = \mathbf{C} \times_{i=1}^{d} U_i$ be a tensor given in Tucker format, and assume, without loss of generality, that $r_1 \geq r_i$, for all i. Then $\mathbf{A}$ can be expressed in the CP format as

$$\mathbf{A} = \sum_{i_2=1}^{r_2} \cdots \sum_{i_d=1}^{r_d} \Big(\sum_{i_1=1}^{r_1} c_{i_1 \cdots i_d} \mathbf{u}_{i_1}^{(1)} \Big) \circ \mathbf{u}_{i_2}^{(2)} \circ \cdots \circ \mathbf{u}_{i_d}^{(d)} \tag{2.14}$$

This shows that the tensor rank is bounded by $\prod_{k=2}^{d} r_k$—a number which grows exponentially in d. For a further discussion and ways of approximating the expression (2.14), see [32, Subsection 8.5.3].

It should be noted that various other low-rank formats for tensors exist, many of which are based on the CP or the Tucker format, see [56, Section 5]. In recent years, two new formats, which avoid the detrimental properties of the CP decomposition while giving polynomial storage and computational complexity, have attracted considerable interest in the literature. One is the *hierarchical Tucker* format [30, 33], which builds a binary tree for the tensor modes to store information recursively. The other is the *tensor train* (*TT*) format [74] (also known as *matrix product states* or *linear tensor network*, especially in physics), which allows the computation of each tensor entry as a product of matrices of small size.

2.2 Computing Low-Rank Tensor Decompositions

In this section, we will introduce some standard algorithms to compute the CP and the Tucker decomposition of a given tensor. We will also discuss the approximation qualities of the result with respect to the full tensor. Our exposition is largely based on [47, Chapter 2].

In this section, as well as in some following parts of the thesis, some special matrix operations will be useful. We will introduce them here briefly.

The *Kronecker product* of two matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$ is denoted by $A \otimes B$. The result is an $mp \times nq$ matrix and defined by

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix}.$$

The *Khatri-Rao product* contains the Kronecker products of the matching columns. Given matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times n}$, their Khatri-Rao is denoted by $A \circledast B$. The result is an $mp \times n$ matrix and defined by

$$A \circledast B = \begin{bmatrix} \mathbf{a}_1 \otimes \mathbf{b}_1 & \dots & \mathbf{a}_n \otimes \mathbf{b}_n \end{bmatrix}.$$

If $\mathbf{a}$, and $\mathbf{b}$ are vectors, then their Khatri-Rao and Kronecker products are identical, i. e. $\mathbf{a} \otimes \mathbf{b} = \mathbf{a} \circledast \mathbf{b}$.

The *Hadamard product* is the elementwise matrix product. Given matrices A and B, both of size $m \times n$, their Hadamard product is denoted by $A \odot B$. The result is also an $m \times n$ matrix and defined by

$$A \odot B = \begin{bmatrix} a_{11}b_{11} & \dots & a_{1n}b_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1}b_{m1} & \dots & a_{mn}b_{mn} \end{bmatrix}.$$

Now, we will collect some useful properties of these operations, cf. [56, Subsection 2.6] Let A, B, C and D matrices with the right dimensions, such that all the operations given below are defined. Then

$$\begin{aligned} (A \otimes B)(C \otimes D) &= AC \otimes BD, && (2.15)\\ (A \otimes B)^+ &= A^+ \otimes B^+, \\ A \circledast B \circledast C &= (A \circledast B) \circledast C = A \circledast (B \circledast C), \\ (A \circledast B)^{\mathrm{T}}(A \circledast B) &= (A^{\mathrm{T}}A) \otimes (B^{\mathrm{T}}B), \\ (A \circledast B)^+ &= \big((A^{\mathrm{T}}A) \otimes (B^{\mathrm{T}}B)\big)^+ (A \circledast B)^{\mathrm{T}}, && (2.16) \end{aligned}$$

where A^+ denotes the *Moore-Penrose pseudoinverse* of A. For more matrix properties of this kind, see, for example, the collection [69].

Now we present some basic algorithms: we consider the CP decomposition in Subsection 2.2.1 and the Tucker decomposition in Subsection 2.2.2.

2.2.1 The Alternating Least Squares Algorithm for the CP Decomposition

In this subsection, we will present the *alternating least squares method* for computing a CP approximation to a given full tensor (CP-ALS). It goes back to the original papers [17, 36] and remains one of the main algorithms for the computation of CP decomposition to this day. It is simple to understand and implement, and the idea of alternating optimization appears at many places in tensor numerical methods, as we will see in the following subsection and also in Chapter 5.

The goal is to solve the optimization problem

$$\begin{aligned} &\min_{\mathbf{X}} \|\mathbf{A} - \mathbf{X}\|^2 \\ &\text{s. t. } \mathbf{X} = [\![\boldsymbol{\lambda}; U, V, W]\!] = \sum_{k=1}^{r} \lambda_k \, \mathbf{u}_k \circ \mathbf{v}_k \circ \mathbf{w}_k. \end{aligned} \tag{2.17}$$

This means, for a given full tensor $\mathbf{A}$ and a fixed r, we want to find the rank-r tensor $\mathbf{X}$, which is closest to $\mathbf{A}$ (if it exists).

For simplicity, we will limit ourselves here to the case of of mode-3 tensors. The d-mode case is completely analogous, but makes notation slightly more cumbersome, see [56, Fig. 3.3].

Using the properties listed in the introduction to this section, we can write the three different matricizations of a tensor $\mathbf{X} = [\![U, V, W]\!]$ as

$$\begin{aligned} X_{(1)} &= U(W \circledast V)^{\mathrm{T}}, \\ X_{(2)} &= V(W \circledast U)^{\mathrm{T}}, \\ X_{(3)} &= W(V \circledast U)^{\mathrm{T}}. \end{aligned}$$

Therefore, the objective function from (2.17) can be written in matricized form:

$$\begin{aligned} \|\mathbf{A} - \mathbf{X}\|^2 &= \|A_{(1)} - U(W \circledast V)^{\mathrm{T}}\|^2 \\ &= \|A_{(2)} - V(W \circledast U)^{\mathrm{T}}\|^2 \\ &= \|A_{(3)} - W(V \circledast U)^{\mathrm{T}}\|^2. \end{aligned}$$

While the objective function is nonlinear when viewed as function of the three factors U, V and W, it is linear in each of the factors when the other two are fixed. This leads us to the alternating method, where we solve an alternating sequence of linear least squares problems. In the mode-1 matricization problem, for V and W fixed, we solve

$$\min_{\hat{U}} \|A_{(1)} - \hat{U}(W \circledast V)^{\mathrm{T}}\|^2,$$

where $\hat{U} = U \operatorname{diag}(\boldsymbol{\lambda})$ contains the weight vector $\boldsymbol{\lambda}$. Since this is just a linear least-squares problem, the solution is given by

$$\hat{U} = A_{(1)}\big((W \circledast V)^{\mathrm{T}}\big)^{+}.$$

Because of the formula for the pseudoinverse of the Khatri-Rao product in (2.16), we can rewrite this solution as

$$\hat{U} = A_{(1)}(W \circledast V)(W^{\mathrm{T}}W \odot V^{\mathrm{T}}V)^{+}.$$

Hence it is possible to solve the least-squares problem for a small square $r \times r$ matrix instead of a tall $n_1 \times r$ matrix, but the condition number is squared—the structure of the last formula is directly related to the normal equations.

Finally, we normalize the columns of $\hat{U}$ to get U and store the column norms in the vector $\boldsymbol{\lambda}$. The computation of the factor matrices V and W is completely analogous. Then, the iterative scheme is repeated until a convergence criterion is met or the maximum number of iterations is exceeded.

Due to its simplicity, the CP-ALS algorithm has remained a widely applied method over the decades, and is still competitive in some practical applications, see, for example, the comparison in [83]. Various modifications have been proposed to the CP-ALS method, but theoretical understanding remains incomplete—notably, it is not guaranteed to converge to a minimum, or even a stationary point, since the full information in all modes is never considered simultaneously. Due to the inherent potential ill-posedness of the best rank-r approximation problem, theoretical progress in this area has been slow. Most recently, a Riemannian optimization approach [14] has been proposed, which might provide new insights.

2.2.2 The HOSVD and Tucker-ALS for Computing the Tucker Decomposition

In this subsection, we will discuss the computation of the Tucker decomposition and the problem of the best multilinear rank-$\mathbf{r}$ approximation to a full tensor. Specifically, we consider the problem

$$\begin{aligned} &\min_{\mathbf{X}} \|\mathbf{A} - \mathbf{X}\|^2 \\ &\text{s. t. } \mathbf{X} = \mathbf{C} \bigtimes_{i=1}^{d} U_i \end{aligned} \tag{2.18}$$

The fact that the multilinear rank of a tensor is based directly on the well-understood concept of the matrix rank gives the motivation for the first approach to the problem (2.18): due to the Eckart–Young approximation theorem [23], the best rank-r approximation to a matrix is given by its truncated SVD.

This concept can be generalized to tensors by the *truncated higher-order SVD* (*HOSVD*) [64]. We will quote the main result of the original paper, which summarizes the properties of the HOSVD:

Theorem 2.1 (HOSVD, [64, Theorem 2])**.** *Every tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ can be written as the product*

$$\mathbf{A} = \mathbf{C} \times_1 U_1 \times \cdots \times_d U_d = \mathbf{C} \bigtimes_{i=1}^{d} U_i, \tag{2.19}$$

in which

i) all $[\mathbf{u}_1^{(i)} \cdots \mathbf{u}_{r_i}^{(i)}] = U_i \in \mathbb{R}^{n_i \times r_i}$ *have orthonormal columns;*

ii) all matricizations $C_{(i)}$ *of the tensor* $\mathbf{C}$ *satisfy the following properties:*

a) all-orthogonality: *the rows of* $C_{(i)}$ *are pairwise orthogonal, i. e.*

$$C_{(i)} C_{(i)}^{\mathrm{T}} =: \Sigma_i$$

is a diagonal matrix.

b) ordering: *the diagonal entries* $(\sigma_1^{(i)})^2, \ldots, (\sigma_{r_i}^{(i)})^2$ *of* Σ_i *satisfy*

$$\sigma_1^{(i)} \geq \cdots \geq \sigma_{r_i}^{(i)} \geq 0.$$

The decomposition (2.19) *is called the* higher-order singular value decomposition *(*HOSVD*) of the tensor* $\mathbf{A}$*. The non-negative numbers* $\sigma_k^{(i)}$ *are called the* i-mode singular values *of* $\mathbf{A}$*, and the vector* $\mathbf{u}_k^{(i)}$ *is the corresponding* i-mode left singular vector *of* $\mathbf{A}$*.*

Proof. Application of the SVD to the matricization of $\mathbf{A}$ in each mode. □

It is worth noting that the HOSVD inherits the uniqueness properties of the matrix SVD: the singular vectors to a distinct singular value are unique up to the sign; the left singular vectors to a multiple singular value are unique up to an orthogonal transformation, see [64, Property 4].

Now, the HOSVD can be applied in truncated form to give a rank-$\mathbf{r}$ approximation to a full tensor. Let $\mathrm{P}_{r_i}^i$ be the best rank-r_i approximation operator in the ith mode, i. e. $\mathrm{P}_{r_i}^i \mathbf{A} = (U_i U_i^{\mathrm{T}} A_{(i)})^{(i)}$, where U_i denotes the matrix of the r_i dominant left singular vectors of $A_{(i)}$. Then the rank-$\mathbf{r}$ truncated HOSVD operator $\mathrm{P}_{\mathbf{r}}^{\mathrm{HO}}$ is given by

$$\mathrm{P}_{\mathbf{r}}^{\mathrm{HO}} \mathbf{A} \coloneqq \mathrm{P}_{r_1}^1 \cdots \mathrm{P}_{r_d}^d \mathbf{A}. \tag{2.20}$$

In contrast to the matrix case, the HOSVD does not yield a best rank-$\mathbf{r}$ approximation: the Eckart–Young result is not applicable in general since the singular values in the different modes do not coincide—except for $d = 2$, where the nonzero singular values of A and A^{T} are equal. If $\mathbf{A}$ is a rank-$\mathbf{s}$ tensor (with $\mathbf{s} \geq \mathbf{r}$, entrywise), the truncation error satisfies

$$\|\mathbf{A} - \mathrm{P}_{\mathbf{r}}^{\mathrm{HO}} \mathbf{A}\| \geq \sum_{i_1=r_1+1}^{s_1} (\sigma_{i_1}^{(1)})^2 + \cdots + \sum_{i_d=r_d+1}^{s_d} (\sigma_{i_d}^{(d)})^2. \tag{2.21}$$

Hence, the HOSVD gives only a quasi-best-approximation with a constant which deteriorates with respect to the number of modes:

$$\left\|\mathbf{A} - \mathrm{P}_{\mathbf{r}}^{\mathrm{HO}} \mathbf{A}\right\| \leq \sqrt{d} \min_{\mathbf{X} \in \mathcal{M}_{\mathbf{r}}} \|\mathbf{A} - \mathbf{X}\|, \tag{2.22}$$

see [64, Property 10].

Since the truncated HOSVD does not give a best rank-$\mathbf{r}$ approximation, there are methods in the literature concerned with improving on the approximation quality of $\mathrm{P}_{\mathbf{r}}^{\mathrm{HO}}\,\mathbf{A}$. They are based on the following theorem (cf. [65]).

Theorem 2.2. *For a given tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, the minimization problem* (2.18) *is equivalent to the dual maximization problem*

$$\max_{V_1,\ldots,V_d} \left\| \mathbf{A} \bigtimes_{i=1}^{d} V_i^{\mathrm{T}} \right\|,$$

where the matrices V_i have full rank. For given maximizing matrices V_i, the core tensor $\mathbf{C}$ minimizing (2.18) *is represented by*

$$\mathbf{C} = \mathbf{A} \bigtimes_{i=1}^{d} V_i^{\mathrm{T}}.$$

Now we can, starting from the HOSVD, build a method for computing the best rank-r approximation to $\mathbf{A}$ in Algorithm 2.3. Note, however, that it does us not, in general, give a guarantee that we find the global optimum.

Algorithm 2.3 Tucker decomposition via ALS.

Input: Input tensor $\mathbf{A}$, multilinear rank $\mathbf{r}$, maximum number of ALS iterations $k_{\max} \geq 1$.

1: Compute the HOSVD of $\mathbf{A}$ for an initial guess, i. e.

$$(\mathbf{C}^{(0)}, U_1^{(0)}, \ldots, U_d^{(0)}) \leftarrow \mathrm{P}_{\mathbf{r}}^{\mathrm{HO}}\,\mathbf{A}$$

2: **for** $k = 1$ **to** $k_{\max}$ **do**
3: **for** $i = 1$ **to** d **do**
4: $\mathbf{B} \leftarrow \mathbf{A} \times_{j=1}^{i-1} \left(U_j^{(k)}\right)^{\mathrm{T}} \times_{j=i+1}^{d} \left(U_j^{(k-1)}\right)^{\mathrm{T}}$
5: $U_i^{(k)} \leftarrow (r_i$ dominant left singular vectors of $B_{(i)})$
6: **end for**
7: **end for**
8: $\mathbf{C} \leftarrow \mathbf{A} \times_{j=1}^{d} \left(U_j^{(k_{\max})}\right)^{\mathrm{T}}$

3 Elements of Riemannian Optimization

As we will see in the next chapter, the set of tensors of fixed multilinear rank forms a smooth submanifold of the space of all tensors; similar properties hold for other fixed-rank sets of tensors. Due to the centrality of optimization on smooth manifolds to this thesis and to the field of tensor methods in general, in this chapter, we will present some basic definitions and methods from this field. We will only introduce the concepts that will be of most use to us later—a comprehensive introduction to the field of optimization on manifolds is far beyond the scope of this thesis; we recommend the standard text [2], which is the basis for this chapter, and the references therein.

In Section 3.1, we will introduce basic concepts of differential geometry needed for the study of manifolds. In Section 3.2, we will utilize these concepts to construct optimization methods on Riemannian manifolds.

3.1 Basic Definitions

In this section we will introduce the definition of the manifold—a basic structure of differential geometry and the "workhorse" of Riemannian optimization.

In Subsection 3.1.1 we will describe what an abstract manifold is, from the point of view of differential geometry. In Subsection 3.1.2, we will describe the tangent space structure of manifolds, which is needed to introduce differential calculus. Finally, in Subsection 3.1.3, we will introduce the Riemannian structure on manifolds, which is a generalization of the Euclidean structure in vector spaces.

3.1.1 Manifolds, Submanifolds, Product and Quotient Manifolds

Intuitively speaking, a manifold $\mathcal{M}$ is a set which is locally similar to the vector space $\mathbb{R}^n$. The abstract definition of a manifold relies on the concept of a chart—a function which maps a region of $\mathcal{M}$ it to $\mathbb{R}^n$ while satisfying certain properties—and an atlas—a collection of maps.

Let $\mathcal{M}$ be a set. A mapping φ of a subset $\mathcal{U}$ of $\mathcal{M}$ onto an open subset of $\mathbb{R}^n$ is called an n-dimensional *chart of the set* $\mathcal{M}$, denoted by $(\mathcal{U}, \varphi)$. Given a chart $(\mathcal{U}, \varphi)$ and $x \in \mathcal{U}$, the entries of $\varphi(x) \in \mathbb{R}^n$ are called the *coordinates* of x in the chart $(\mathcal{U}, \varphi)$.

The concept of a chart makes it possible to apply the usual methods of real analysis to objects associated wit $\mathcal{U}$. For example, if f is a real-valued function on $\mathcal{U}$, then $f \circ \varphi^{-1}$ is a function form $\mathbb{R}^n$ to $\mathbb{R}$, with domain $\varphi(\mathcal{U})$. To apply this idea to the whole set $\mathcal{M}$, we must ensure that each point $x \in \mathcal{M}$ belongs to at least one chart domain; however, if x belongs to the domains of two charts $(\mathcal{U}_1, \varphi_1)$ and $(\mathcal{U}_2, \varphi_2)$, then the two charts must

give compatible information with respect to differentiability. This leads to the concept of the atlas, which will help us define a manifold.

Definition 3.1 (atlas, manifold)**.** A *(C^∞) atlas of $\mathcal{M}$ into $\mathbb{R}^n$* is a collection of charts $(\mathcal{U}_\alpha, \varphi_\alpha)$ of the set $\mathcal{M}$ such that

i) $\bigcup_\alpha \mathcal{U}_\alpha = \mathcal{M}$,

ii) for any pair α, β with $\mathcal{U}_\alpha \cap \mathcal{U}_\beta \neq \varnothing$, the sets $\varphi_\alpha(\mathcal{U}_\alpha \cap \mathcal{U}_\beta)$ and $\varphi_\beta(\mathcal{U}_\alpha \cap \mathcal{U}_\beta)$ are open sets in $\mathbb{R}^n$ and the change of coordinates

$$\varphi_\beta \circ \varphi_\alpha^{-1} : \mathbb{R}^n \to \mathbb{R}^n$$

is *smooth* (i. e. of class C^∞) on its domain $\varphi_\alpha(\mathcal{U}_\alpha \cap \mathcal{U}_\beta)$. We say that the elements of an atlas *overlap smoothly.*

Given an atlas $\mathcal{A}$, let $\mathcal{A}^+$ be the set of all charts $(\mathcal{U}, \varphi)$, such that $\mathcal{A} \cup \{(\mathcal{U}, \varphi)\}$ is also an atlas. Then $\mathcal{A}^+$ is also an atlas, called the *maximal atlas* (or *complete atlas*) generated by the atlas $\mathcal{A}$.

An *(n-dimensional) manifold* is a couple $(\mathcal{M}, \mathcal{A}^+)$, where $\mathcal{M}$ is a set and $\mathcal{A}^+$ is a maximal atlas of $\mathcal{M}$ into $\mathbb{R}^n$, such that the topology induced by $\mathcal{A}^+$ is Hausdorff and second-countable.

A maximal atlas of a set $\mathcal{M}$ that induces a second-countable Hausdorff topology is called a *manifold structure* on $\mathcal{M}$. Given a chart φ on $\mathcal{M}$, the inverse mapping φ^{-1} is called a *local parametrization* of $\mathcal{M}$.

A few remarks to the Definition 3.1 are in order:

i) The definition of the manifold contains two technical assumptions: We assume that the induced topology is *Hausdorff*, i. e. any two distinct points of $\mathcal{M}$ have disjoint neighbourhoods, and that it is *second-countable*, i. e. there is a countable collection $\mathcal{B}$ of open sets such that every open set is the union of some subcollection of $\mathcal{B}$. These assumptions ensure that basic results from differential geometry hold and preclude counterintuitive properties of $(\mathcal{M}, \mathcal{A}^+)$. More details may be found in [2, Subsection 3.1.2] and in standard texts on differential geometry, such as [85].

ii) If $(\mathcal{M}, \mathcal{A}^+)$ is a manifold, we will usually drop the atlas and simply say "the manifold $\mathcal{M}$" when there is no ambiguity.

iii) A family of local parametrizations $(\varphi_\alpha^{-1})_\alpha$ is equivalent to a family of charts $(\varphi_\alpha)_\alpha$, and the definition of a manifold may be given in terms of either.

Given two manifolds, one can easily construct a new one. Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be manifolds of dimension n_1 and n_2, respectively. If $(\mathcal{U}_1, \varphi_1)$ and $(\mathcal{U}_2, \varphi_2)$ are charts of the manifolds $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively, then the mapping

$$\varphi_1 \times \varphi_2 : \mathcal{U}_1 \times \mathcal{U}_2 \to \mathbb{R}^{n_1} \times \mathbb{R}^{n_2},\ (x_1, x_2) \mapsto (\varphi_1(x_1), \varphi_2(x_2))$$

is a chart for the set $\mathcal{M}_1 \times \mathcal{M}_2$. All the charts thus obtained form an atlas for the set $\mathcal{M}_1 \times \mathcal{M}_2$. With the structure defined by this atlas, $\mathcal{M}_1 \times \mathcal{M}_2$ is called the *product* of the manifolds $\mathcal{M}_1$ and $\mathcal{M}_2$. Its manifold topology is equivalent to the product topology.

Many tensor manifolds, including the one studied in the next chapter, admit a submanifold structure, i. e. they are embedded into a larger space. Generally speaking, let $(\mathcal{M}, \mathcal{A}^+)$ and $(\mathcal{N}, \mathcal{B}^+)$ be manifolds such that $\mathcal{N} \subset \mathcal{M}$. The manifold $(\mathcal{N}, \mathcal{B}^+)$ is called an *immersed submanifold* of $(\mathcal{M}, \mathcal{A}^+)$ if the inclusion map $i : \mathcal{N} \to \mathcal{M}$, $x \mapsto x$, is an *immersion*, i. e. the derivative $\mathrm{D}\, i(x)$ is an injection for all x. If the manifold topology of $\mathcal{N}$ coincides with its subspace topology induced from the topological space $\mathcal{M}$, then $\mathcal{N}$ is called an *embedded submanifold*, or simply a *submanifold* of the manifold $\mathcal{M}$. An important special case is that of $\mathcal{M}$ being a vector space.

It is important to note that, for any subset $\mathcal{N}$ of a manifold $\mathcal{M}$, there is at most one atlas, which makes it an embedded submanifold of $\mathcal{M}$, cf. [2, Proposition 3.3.1]. Therefore, we can always speak about submanifolds without any ambiguity about the underlying differentiable structure.

In practice, checking the definition of a submanifold can be difficult. Therefore, we quote two important sufficient conditions for subsets of manifolds to be embedded submanifolds.

Lemma 3.2 (submersion theorem, subimmersion theorem, [2, Propositions 3.3.3, 3.3.4])**.** *Let $F : \mathcal{M}_1 \to \mathcal{M}_2$ be a smooth mapping between two manifolds of dimension n_1 and n_2, respectively.*

i) Submersion theorem: *Let $n_1 > n_2$ and let y be a point of $\mathcal{M}_2$. If the rank of F (i. e. the rank of $\mathrm{D}\, F$)is equal to n_2 at evry point of $F^{-1}(\{y\})$, then $F^{-1}(\{y\})$ is a closed embedded submanifold of $\mathcal{M}_1$, and $\dim(F^{-1}(\{y\})) = n_1 - n_2$.*

ii) Subimmersion theorem: *Let y be a point of $F(\mathcal{M}_1)$. If F has constant rank $k < n_1$ in a neighbourhood of $F^{-1}(\{y\})$, then $F^{-1}(\{y\})$ is a closed embedded submanifold of $\mathcal{M}_1$ of dimension $n_1 - k$.*

We will conclude this subsection with the concept of quotient manifolds—these are manifolds whose elements are equivalence classes. This concept will be useful when studying objects, which have several equivalent representations—such as tensors of fixed multilinear rank—since it will be sufficient to consider any representative of a class.

Let $\mathcal{M}$ be a manifold equipped with an *equivalence relation* $\sim$, i. e. a relation that is

i) reflexive: $x \sim x$, for all $x \in \mathcal{M}$,

ii) symmetric: $x \sim y$ if and only if $y \sim x$, for all $x, y \in \mathcal{M}$,

iii) transitive: if $x \sim y$ and $y \sim z$, then $x \sim z$, for all $x, y, z \in \mathcal{M}$.

The set

$$[x] \coloneqq \{y \in \mathcal{M} \,|\, y \sim x\}$$

of all elements that are equivalent to a a point x is called the *equivalence class* containing x. The set

$$\mathcal{M}/\sim := \{[x] \in \mathcal{M} \,|\, x \sim x\}$$

of all equivalence classes of $\sim$ in $\mathcal{M}$ is called the *quotient* of $\mathcal{M}$ by $\sim$. The mapping $\pi : \mathcal{M} \to \mathcal{M}/\sim,\ x \mapsto [x]$, which maps each element of $\mathcal{M}$ to its equivalence class, is called the *natural* or *canonical projection*.

Now let $(\mathcal{M}, \mathcal{A}^+)$ be a manifold with an equivalence relation $\sim$ and let $\mathcal{B}^+$ be a manifold structure on the set $\mathcal{M}/\sim$. The manifold $(\mathcal{M}/\sim, \mathcal{B}^+)$ is called a *quotient manifold* of $(\mathcal{M}, \mathcal{A}^+)$ if the natural projection π is a submersion, i. e. if its derivative is surjective at each point in $\mathcal{M}$.

Similarly to the case of submanifolds, a quotient manifold structure is unique. This means, if $\mathcal{M}$ is a manifold and $\mathcal{M}/\sim$ is a quotient of $\mathcal{M}$, then $\mathcal{M}/\sim$ admits at most one manifold structure that makes it a quotient manifold of $\mathcal{M}$, cf. [2, Proposition 3.4.1].

3.1.2 Tangent Vectors and Derivatives

Since a manifold is locally similar to $\mathbb{R}^n$, we can use tools of real analysis to work with functions on manifolds. However, we will need the notion of tangent vectors and the tangent space of a manifold to approach functions on manifolds in this way. We will start with the fundamental definition of this subsection.

Definition 3.3 (curve, tangent vector, tangent space, tangent bundle)**.** Let $\mathcal{M}$ be a manifold.

i) A smooth mapping

$$\gamma : \mathbb{R} \to \mathcal{M},\ t \mapsto \gamma(t)$$

is called a *curve in* $\mathcal{M}$.

ii) Let $x \in \mathcal{M}$, and denote the set of all smooth, real-valued functions defined on a neighbourhood of x by $\mathcal{F}_x(\mathcal{M})$. A *tangent vector* ξ_x $\mathcal{M}$ at x is a mapping from $\mathcal{F}_x(\mathcal{M})$ to $\mathbb{R}$ such that there exists a curve γ on $\mathcal{M}$ with $\gamma(0) = x$, satisfying

$$\xi_x f = \dot{\gamma}(0) f := \left.\frac{\mathrm{d}(f(\gamma(t)))}{\mathrm{d}t}\right|_{t=0},$$

for all $f \in \mathcal{F}_x(\mathcal{M})$. Such a curve γ is said to *realize* the tangent vector ξ_x. The point x is called the *foot* of the tangent vector ξ_x.

iii) The *tangent space* to $\mathcal{M}$ at a point x, denoted by $T_x\mathcal{M}$ is the set of all tangent vectors to $\mathcal{M}$ at x.

iv) The set of all tangent vectors to $\mathcal{M}$, denoted by

$$T\mathcal{M} := \bigcup_{x \in \mathcal{M}} T_x\mathcal{M},$$

is called the *tangent bundle* of $\mathcal{M}$.

Again, some remarks are in order concerning this definition:

i) When there is no ambiguity, we will occasionally simply write $\xi = \xi_x$.

ii) The tangent vector is well defined, i. e. the definition does not depend on the choice of the curve γ. Similarly, the linear structure of the vector space $T_x\mathcal{M}$ is well defined. Details can be found in [2, Subsection 3.5.1].

iii) It can be shown that the tangent bundle $T\mathcal{M}$ itself is a manifold, cf. [2, Subsection 3.5.4].

The concept of tangent vectors allows us now to define derivatives on manifolds. Let $F : \mathcal{M} \to \mathcal{N}$ be a smooth mapping between two manifolds $\mathcal{M}$ and $\mathcal{N}$. Let ξ_x be a tangent vector at a point x of $\mathcal{M}$. It can be shown that the mapping $\mathrm{D}\,F(x)[\xi_x]$ from $\mathcal{F}_x(\mathcal{N})$ to $\mathbb{R}$ defined by

$$(\mathrm{D}\,F(x)[\xi_x])f \coloneqq \xi_x(f \circ F)$$

is a tangent vector to $\mathcal{N}$ at $F(x)$. The tangent vector $\mathrm{D}\,F(x)[\xi_x])$ is realized by $F \circ \gamma$, where γ is any curve that realizes ξ_x. The mapping

$$\mathrm{D}\,F(x) : T_x\mathcal{M} \to T_{F(x)}\mathcal{N},\ \xi_x \mapsto \mathrm{D}\,F(x)[\xi_x]$$

is a linear mapping called the *differential* (or *differential map*, *derivative*, or *tangent map*) of F at x.

If $\mathcal{N} = \mathbb{R}$, i. e. F is a real-valued function on a manifold $\mathcal{M}$, then $F \in \mathcal{F}_x(\mathcal{M})$, and we simply have

$$\mathrm{D}\,F(x)[\xi_x] = \xi_x F$$

using the fact that the tangent space to $\mathbb{R}$ is $\mathbb{R}$ itself.

We have seen that, on general manifolds, a tangent vector is an object that, given a real-valued function, returns a real number. Now let $\mathcal{M}$ be an embedded submanifold of a vector space $\mathcal{E}$. Let γ be a curve in $\mathcal{M}$ with $\gamma(0) = x$. Let $\bar{f}$ denote a real-valued function in a neighbourhood $\mathcal{U}$ of x in $\mathcal{E}$ and let $f = \bar{f}|_{\mathcal{U}\cap\mathcal{M}}$. Then we have

$$\dot{\gamma}(0)f = \frac{\mathrm{d}}{\mathrm{d}t} f(\gamma(t))\bigg|_{t=0} = \frac{\mathrm{d}}{\mathrm{d}t} \bar{f}(\gamma(t))\bigg|_{t=0} = \mathrm{D}\,\bar{f}(x)[\dot{\gamma}(0)].$$

This yields a natural identification of $T_x\mathcal{M}$ with the set

$$\{\dot{\gamma}(0) \,|\, \gamma \text{ curve in } \mathcal{M} \text{ with } \gamma(0) = x\},$$

which is a linear subspace of the vector space $\mathcal{E}$. Graphically, a tangent vector to a submanifold of a vector space can be thought of as an "arrow" tangent to the manifold. The tangent space can be thought of as the plane touching the manifold in one point. See Figure 3.1 for an illustration.

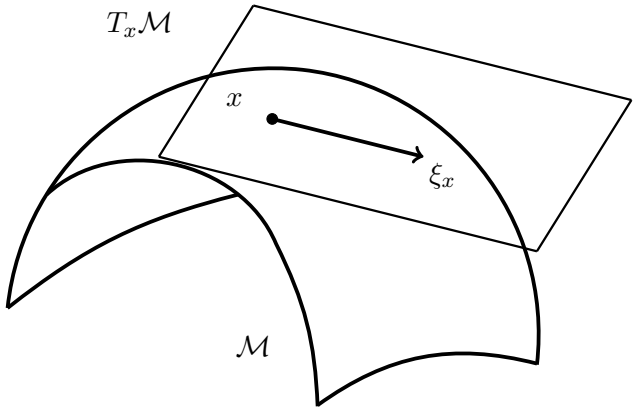


Figure 3.1: Tangent space of an embedded submanifold of a vector space.

3.1.3 Riemannian Metric and Gradients

In the previous subsection, we have defined directional derivatives on manifolds. In order to define a gradient as the representative vector of the derivative, we will need the notion of an inner product. Since we have a distinct tangent vector space $T_x\mathcal{M}$ in each point x of a manifold, we will naturally have an inner product $\langle \cdot, \cdot \rangle_x$ for every $T_x\mathcal{M}$. It induces a norm $\|\xi_x\|_x \coloneqq \sqrt{\langle \xi_x, \xi_x \rangle_x}$.

A manifold whose tangent spaces are endowed with a smoothly varying inner product is called a *Riemannian manifold*. The smoothly varying inner product is called the *Riemannian metric*. When there is no ambiguity, we will drop the index x from the inner product and the norm.

A *metric* on $\mathcal{M}$ (which should not be confused with the Riemannian metric on $T_x\mathcal{M}$) can be defined with the help of curves. The *length of a curve* $\gamma : [a, b] \to \mathcal{M}$ is defined by

$$L(\gamma) \coloneqq \int_a^b \left\| \dot{\gamma}(t) \right\| \mathrm{d}t.$$

Then the *Riemannian distance* on $\mathcal{M}$ is given by

$$\operatorname{dist} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}, \ \operatorname{dist}(x, y) = \inf_{\gamma \in \Gamma} L(\gamma),$$

where Γ is the set of all curves in $\mathcal{M}$ joining the points x and y. Hence, intuitively, the distance between x and y is equal to the length of the shortest connection between them.

Strictly speaking, a Riemannian manifold is a couple $(\mathcal{M}, \langle \cdot, \cdot \rangle)$ of a manifold and a Riemannian metric. However, when the Riemannian metric is clear, we will drop it and simply speak of $\mathcal{M}$ as a Riemannian manifold. It can be shown that any (second-countable Hausdorff) manifold admits a Riemannian structure.

Given a smooth real-valued function f on a Riemannian manifold $\mathcal{M}$, the *(Riemannian) gradient* of f at x, denoted by $\operatorname{grad} f(x)$ is defined as the unique (due to the Riesz

representation theorem) element of $T_x\mathcal{M}$ that satisfies

$$\langle \operatorname{grad} f(x), \xi \rangle_x = \mathrm{D}\, f(x)[\xi], \text{ for all } \xi \in T_x\mathcal{M}.$$

For the use in optimization methods, it is worth noting that the Riemannian gradient gradient is the steepest ascent direction of f at x, i. e.

$$\frac{\operatorname{grad} f(x)}{\|\operatorname{grad} f(x)\|} = \operatorname*{arg\,max}_{\xi \in T_x\mathcal{M},\, \|\xi\|=1} \mathrm{D}\, f(x)[\xi]. \tag{3.1}$$

Again, we will discuss the special case of embedded submanifolds, which is of particular interest to this thesis. Let $\mathcal{M}$ be an embedded submanifold of a Euclidean vector space $\mathcal{E}$. Since every tangent space $T_x\mathcal{M}$ can be regarded as a subspace of $\mathcal{E}$, the inner product on $\mathcal{E}$ induces a Riemannian metric on $\mathcal{M}$, simply by the natural inclusion map. This turns $\mathcal{M}$ into a Riemannian manifold. Endowed with this Riemannian metric, $\mathcal{M}$ is called a *Riemannian submanifold* of $\mathcal{E}$. The orthogonal complement of $T_x\mathcal{M}$ is called the *normal space* to $\mathcal{M}$ at x and is denoted by

$$(T_x\mathcal{M})^\perp = \{\xi \in \mathcal{E} \mid \langle \xi, \eta \rangle = 0, \text{ for all } \eta \in T_x\mathcal{M}\}.$$

Any element $\xi \in \mathcal{E}$ can be uniquely decomposed into the sum of an element of $T_x\mathcal{M}$ and an element of $(T_x\mathcal{M})^\perp$:

$$\xi = \mathrm{P}_x\, \xi + \mathrm{P}_x^\perp\, \xi,$$

where P_x denotes the orthogonal projection onto $T_x\mathcal{M}$, and $\mathrm{P}_x^\perp$ denotes the orthogonal projection onto $(T_x\mathcal{M})^\perp$.

Finally, for Riemannian submanifolds, we will present a result, which allows to compute gradients, when given a gradient in the embedding space.

Lemma 3.4 (gradients on submanifolds, [2, Section 3.6.1])**.** *Let $\mathcal{M}$ be a Riemannian submanifold of a Euclidean space $\mathcal{E}$. Let $\bar{f} : \mathcal{E} \to \mathbb{R}$ be a function with Euclidean gradient* $\operatorname{grad} \bar{f}(x)$ *at point $x \in \mathcal{M}$. Then the Riemannian gradient of $f := \bar{f}|_{\mathcal{M}}$ is given by* $\operatorname{grad} f(x) = \mathrm{P}_x \operatorname{grad} \bar{f}(x)$.

3.2 Riemannian Optimization Methods

Now that we have defined all the basic notions from differential geometry needed to work on Riemannain manifolds, we can approach optimization methods on Riemanniain manifolds. In Subsection 3.2.1, we will introduce the basic building blocks of optimization on Riemannian manifolds and present some simple first-order methods. In Subsection 3.2.2, we will introduce the basic second-order method, the Riemannian Newton method. Finally, in Subsection 3.2.3, we will introduce the Riemannian trust-region method, which is a globally convergent second-order method.

3.2.1 Retractions, Vector Transports and Line-Search Methods

Conceptually, the simplest approach to optimizing a differentiable function on a manifold $\mathcal{M}$ is to find a descent direction η_x on some test point x. After a sequence of iterations, we hope to converge to a *critical* or *stationary point* x^*, where $\operatorname{grad} f(x^*) = 0$. Stationarity is a necessary condition for local optimality; we refer to the text by Udrişte [87] for a detailed discussion of optimality conditions on Riemannian manifolds.

In $\mathbb{R}^n$, the concept of moving from a point in the direction of a vector is straightforward. On a manifold, we lack an underlying linear structure for this operation. However, we can move in a direction on the tangent space $T_x\mathcal{M}$ of the point x.

However, a descent direction η_k in x_k does not us give a new iterate x_{k+1} yet, since there is no linear structure we can use to combine x_k with η_k. Here the definition of a retraction on $\mathcal{M}$ comes into play.

Definition 3.5 (retraction, [2, Definition 4.1.1])**.** A *retraction* on a manifold $\mathcal{M}$ is a smooth mapping R from the tangent bundle $T\mathcal{M}$ onto $\mathcal{M}$ with the following properties (where R_x denotes the restriction of R to $T_x\mathcal{M}$).

i) $R_x(0_x) = x$, where 0_x denotes the zero element of $T_x\mathcal{M}$.

ii) With the canonical identification $T_{0_x}T_x\mathcal{M} \simeq T_x\mathcal{M}$, the mapping R_x satisfies the *rigidity condition*

$$\mathrm{D}\, R_x(0_x) = \mathrm{id}_{T_x\mathcal{M}},$$

where $\mathrm{id}_{T_x\mathcal{M}}$ denotes the identity mapping on $T_x\mathcal{M}$.

Now, instead of the expression "$x_k + \eta_k$", we can use $R_x(\eta_k)$ to obtain the new iterate, cf. Figure 3.2. We note that the rigidity condition means that a retraction is a first-order approximation to the so-called *exponential map* on $\mathcal{M}$, which gives a minimal-distance curve on $\mathcal{M}$ and is an analogue of a straight line in a vector space, see [2, Section 5.4] for details. Since derivative-based optimization methods only use local information anyway,

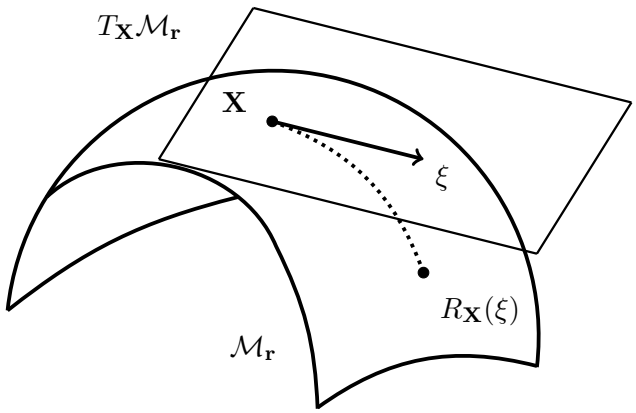


Figure 3.2: Retraction R on a manifold $\mathcal{M}$.

we will find that this kind of local first-order approximation is sufficient to construct convergent methods.

More specifically, since the negative gradient gives us the steepest descent direction (cf. (3.1)), we will want the descent direction η_x to be reasonably close to $-\operatorname{grad} f(x)$ in some sense. Formally, this means: given a cost function f on a Riemannian manifold $\mathcal{M}$, a sequence $(\eta_k)_{k=0}^{\infty}$, $\eta_k \in T_{x_k}\mathcal{M}$, is *gradient-related* for f, if, for any subsequence $(x_{k_i})_{i=0}^{\infty}$ of $(\eta_k)_{k=0}^{\infty}$ that converges to a non-critical point of f, the corresponding subsequence $(\eta_{k_i})_{i=0}^{\infty}$ is bounded and satisfies

$$\limsup_{i\to\infty} \langle \operatorname{grad} f(x_{k_i}), \eta_{k_i} \rangle < 0. \tag{3.2}$$

To construct an optimization method, we will further need a line-search procedure in the tangent space of a manifold $\mathcal{M}$. Given a cost function f on a Riemannian manifold $\mathcal{M}$ with retraction R, a point $x \in \mathcal{M}$, a tangent vector $\eta \in T_x\mathcal{M}$, and scalars $\bar{\alpha} > 0$, $\beta, \sigma \in (0,1)$, the *Armijo point* is

$$\eta^{\mathrm{A}} = t^{\mathrm{A}}\eta = \beta^m \bar{\alpha}\eta,$$

where m is the smallest nonnegative integer such that

$$f(x) - f(R_x(\beta^m \bar{\alpha}\eta)) \geq -\sigma \langle \operatorname{grad} f(x), \beta^m \bar{\alpha}\eta \rangle. \tag{3.3}$$

The factor t^{A} is called *Armijo step size*.

Now we can define a simple framework for line-search methods on a manifold in Algorithm 3.6.

Algorithm 3.6 Line search method on a manifold m $\mathcal{M}_{\mathbf{r}}$

Input: Riemannian manifold $\mathcal{M}$; continuously differentiable real-valued function f on $\mathcal{M}$; retraction R from $T\mathcal{M}$ to $\mathcal{M}$; scalars $\bar{\alpha} > 0$, $c, \beta, \sigma \in (0,1)$; initial iterate x_0.

1: **for** $k = 0$ **until** convergence **do**

2: Pick η_k in $T_{x_k}\mathcal{M}$ such that the sequence $(\eta_k)_{k=0}^{\infty}$ is gradient-related (cf. (3.2))

3: <Test for convergence>

4: Select x_{k+1} such that

$$f(x_k) - f(x_{k+1}) \geq c(f(x_k) - f(R_{x_k}(t^{\mathrm{A}}\eta_k))),$$

where t^{A} is the Armijo step size (cf. (3.3)) for the given parameters $\bar{\alpha}, \beta, \sigma, \eta_k$

5: **end for**

As in the case of Euclidean optimization, standard convergence results can be proved for the generic line-search method.

Theorem 3.7 (accumulation points of the line-search method, [2, Theorem 4.3.1])**.** *Let $(x_k)_{k=0}^{\infty}$ be an infinite sequence of iterates generated by Algorithm 3.6. Then every accumulation point of $(x_k)_{k=0}^{\infty}$ is a critical point of the cost function f.*

Proof. By contradiction, using (3.2) and (3.3). □

More can be said under compactness assumptions:

Corollary 3.8 (convergence to a stationary point, [2, Corollary 4.3.3])**.** *Let $(x_k)_{k=0}^{\infty}$ be an infinite sequence of iterates generated by Algorithm 3.6. Assume that the level set $\{x \,|\, x \in \mathcal{M} f(x) \leq f(x_0)\}$ is compact. Then*

$$\lim_{k\to\infty} \|\operatorname{grad} f(x_k)\| = 0.$$

Proof. By contradiction, using Theorem 3.7. □

For numerical methods, not only the convergence itself, but also the speed of convergence is of importance, which we will define as follows [2, Definitions 4.5.1, 4.5.2]:

Definition 3.9 (linear, superlinear, quadratic convergence)**.** Let $\mathcal{M}$ be a Riemannian manifold and let dist denote the Riemannian distance on $\mathcal{M}$. Let $(x_k)_{k=0}^{\infty}$ be a sequence converging to a point $x^* \in \mathcal{M}$.

i) We say that $(x_k)_{k=0}^{\infty}$ converges *linearly* to a point $x^* \in \mathcal{M}$ if there exists a constant $c \in (0,1)$ and an integer $K \geq 0$ such that, for all $k \geq K$, it holds that

$$\operatorname{dist}(x_{k+1}, x^*) \leq c \operatorname{dist}(x_k, x^*). \tag{3.4}$$

The limit

$$\limsup_{k\to\infty} \frac{\operatorname{dist}(x_{k+1}, x^*)}{\operatorname{dist}(x_k, x^*)}$$

is called the *linear convergence factor* of the sequence. An iterative algorithm on $\mathcal{M}$ is said to *converge locally linearly* to a point x^*, if there exists a neighbourhood $\mathcal{V}$ of x^* and a constant $c \in (0,1)$ such that, for every initial point $x_0 \in \mathcal{V}$, the sequence $(x_k)_{k=0}^{\infty}$ generated by the algorithm satisfies (3.4). The constant c is called the *rate of convergence*.

ii) Let $(\mathcal{U}, \varphi)$ be a chart of $\mathcal{M}$ with $x \in \mathcal{U}$. If

$$\lim_{k\to\infty} \frac{\|\varphi(x_{k+1}) - \varphi(x^*)\|}{\|\varphi(x_k) - \varphi(x^*)\|} = 0,$$

then $(x_k)_{k=0}^{\infty}$ is said to converge *superlinearly* to x^*. If there exist constants $p > 0$, $c \geq 0$, and $K \geq 0$ such that, for all $k \geq K$, it holds that

$$\|\varphi(x_{k+1}) - \varphi(x^*)\| \leq c\|\varphi(x_k) - \varphi(x^*)\|^p, \tag{3.5}$$

then x_k is said to converge to x^* with *order* at least p. An iterative algorithm on $\mathcal{M}$ is said to *converge locally* to a point x^* with *order* at least p if there exists a chart $(\mathcal{V}, \psi)$ at x^* and a constant $c > 0$ such that, for every initial point $x_0 \in \mathcal{V}$, the sequence $(x_k)_{k=0}^{\infty}$ generated by the algorithm satisfies (3.5). If $p = 2$, the convergence is said to be *quadratic*.

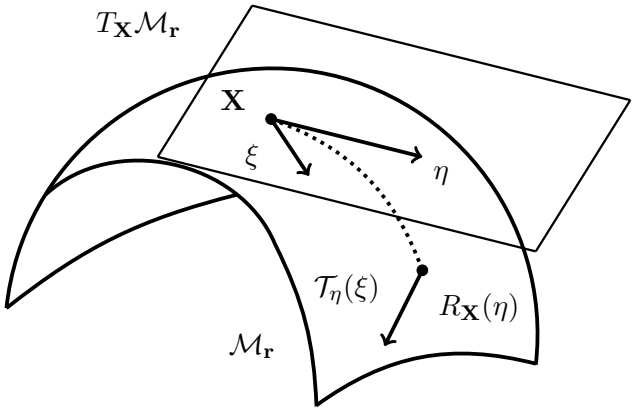


Figure 3.3: Vector transport on a manifold $\mathcal{M}$.

It can be proven that Algorithm 3.6 with $\eta_k = -\operatorname{grad} f(x_k)$ (i. e. the *gradient method* or *steepest descent method*) converges linearly to the limit x^*, cf. [2, Theorem 4.5.6]. However, the rate of convergence depends on the condition number of the Hessian in the critical point x^* and can be bad for large condition numbers, which can lead to very slow convergence in practice, see, for example [2, Subsection 4.6.4], for an illustration.

An improvement over the simple steepest descent method is given by the *nonlinear conjugate gradient* or *CG* method, which uses gradient information from the previous iteration to obtain an improved search direction. However, in the Riemannian case, combining information from two different iterations means combining tangent vectors from two different tangent spaces. This motivates the following definition, see also Figure 3.3.

Definition 3.10 (vector transport, [2, Definition 8.1.1])**.** A *vector transport* on a manifold $\mathcal{M}$ is a smooth mapping

$$\mathcal{T} : T\mathcal{M} \oplus T\mathcal{M} \to T\mathcal{M},\ (\eta, \xi) \mapsto \mathcal{T}_\eta(\xi),$$

satisfying the following properties for all $x \in \mathcal{M}$:

i) (Associated retraction) There exists a retraction R, called the *retraction associated with* $\mathcal{T}$, such that, for all η, ξ, it holds that $\mathcal{T}_\eta \xi \in T_{R_x(\eta)}\mathcal{M}$.

ii) (Consistency) $\mathcal{T}_{0_x}\xi = \xi$ for all $\xi \in T_x\mathcal{M}$.

iii) (Linearity) The mapping $\mathcal{T}_\eta : T_x\mathcal{M} \to T_{R_x(\eta)}\mathcal{M},\ \xi \mapsto \mathcal{T}_\eta \xi$ is linear.

Now we can define the *Riemannian CG* method (also called the *geometric CG* method) in Algorithm 3.11. Following [82, Subsection 2.8.3], we apply the Armijo condition in a backtracking scheme. From the various possibilities for nonlinear CG methods, we choose the *Fletcher–Reeves* update, see, for example [73, Section 5.2], for a discussion. We follow the Riemannian Fletcher–Reeve formula of [2, Section 8.3] and apply a modification,

which ensures that the search sequence $(\eta_k)_{k=0}^{\infty}$ stays sufficiently (in the numerical sense) gradient-related. This means for the coefficient β_k from the CG method:

$$\beta_k = \max\left\{0, \frac{\langle \operatorname{grad} f(x_k), \operatorname{grad} f(x_k) - \mathcal{T}_{\alpha_k \eta_k}(\operatorname{grad} f(x_{k-1}))\rangle}{\|\operatorname{grad} f(x_{k-1})\|^2}\right\}. \tag{3.6}$$

Algorithm 3.11 Riemannian CG method on a manifold $\mathcal{M}$, [82, Algorithm 2.3].

Input: Riemannian manifold $\mathcal{M}$; continuously differentiable real-valued function f on $\mathcal{M}$; vector transport $\mathcal{T}$ with associated retraction R; backtracking parameter $c \in (0,1)$; initial iterate x_0.

1: $\xi_0 \leftarrow \operatorname{grad} f(x_0)$
2: $\eta_0 \leftarrow -\xi_0$
3: $\alpha_0 \leftarrow \arg\min f(R_{x_0}(\alpha\eta_0))$
4: $x_1 \leftarrow R_{x_0}(\alpha_0\eta_0)$
5: **for** $k = 1$ **until** convergence **do**
6: $\quad \xi_k \leftarrow \operatorname{grad} f(x_k)$
7: $\quad \eta_k \leftarrow -\xi_k + \beta_k \mathcal{T}_{\alpha_{k-1}\eta_{k-1}}(\eta_{k-1})$
8: $\quad \alpha_k \leftarrow \arg\min f(R_{x_k}(\alpha\eta_k))$
9: $\quad$ Find smallest $m \geq 0$ such that

$$f(x_k) - f(R_{x_k}(2^{-m}\alpha_k, \eta_k)) \geq -c\langle \xi_k, 2^{-m}\alpha_k\eta_k\rangle$$

10: $\quad x_{k+1} \leftarrow R_{x_k}(2^{-m}\alpha_k\eta_k)$
11: $\quad$ <Test for convergence>
12: **end for**

3.2.2 The Riemannian Hessian and the Newton Method

As we have seen in the previous subsection, steepest descent and CG methods only guarantee linear convergence. To get a superlinearly convergent optimization method on a manifold, we will need information about the second derivative of the objective function. In order to define second derivatives on Riemannian manifolds, we will introduce some basic concepts.

Definition 3.12 (vector field, affine connection, covariant derivative, Lie bracket)**.** Let $\mathcal{M}$ be a manifold with the tangent bundle $T\mathcal{M}$.

i) A *vector field* on $\mathcal{M}$ is a smooth function from $\mathcal{M}$ to $T\mathcal{M}$. Given a vector field ξ on $\mathcal{M}$ and a smoooth real-valued function f on $\mathcal{M}$, we let ξf denote the real-valued function on $\mathcal{M}$ defined by

$$(\xi f)(x) \coloneqq \xi_x(f),$$

for all $x \in \mathcal{M}$. We denote the set of all vector fields on $\mathcal{M}$ by $\mathcal{X}(\mathcal{M})$.

ii) An *affine connection* ∇ on a manifold is a mapping

$$\nabla : \mathcal{X}(\mathcal{M}) \times \mathcal{X}(\mathcal{M}) \to \mathcal{X}(\mathcal{M}),\ (\eta, \xi) \mapsto \nabla_\eta \xi,$$

which satisfies the following properties:

a) $\mathcal{F}(\mathcal{M})$-linearity in η: $\nabla_{f\eta+g\chi}\xi = f\nabla_\eta\xi + g\nabla_\chi\xi$,

b) $\mathbb{R}$-linearity in ξ: $\nabla_\eta(a\xi + b\zeta) = a\nabla_\eta\xi + b\nabla_\eta\zeta$,

c) Product rule (Leibniz's law): $\nabla_\eta(f\xi) = (\eta f)\xi + f\nabla_\eta\xi$,

for any $\eta, \xi, \zeta, \chi \in \mathcal{X}(\mathcal{M})$, $f, g \in \mathcal{F}(\mathcal{M})$, and $a, b \in \mathbb{R}$. The vector field $\nabla_\eta\xi$ is called the *covariant derivative* of ξ with respect to η for the affine connection ∇.

iii) Let ξ and η be vector fields on $\mathcal{M}$. Then we call the function $[\xi, \eta](\,\cdot\,)$ from $\mathcal{F}(\mathcal{M})$ onto itself, defined by

$$[\xi, \eta]f \coloneqq \xi(\eta f) - \eta(\xi f),$$

the *Lie bracket* of ξ and η.

Definition 3.12 allows us now to consider a derivative of a tangent vector ξ in the direction of another tangent vector η. It can be proved (cf. [2, Proposition 5.2.1]) that every manifold admits an affine connection. However, it need not be unique in general. Introducing a Riemannian structure to $\mathcal{M}$ and demanding two more conditions gives us uniqueness.

Theorem 3.13 (Levi–Civita, [2, Theorem 5.3.1])**.** *On a Riemannain manifold $\mathcal{M}$, there exists a unique affine connection ∇ that satisfies*

i) $\nabla_\eta\xi - \nabla_\xi\eta = [\xi, \eta]$ *(symmetry), and*

ii) $\chi\langle\eta, \xi\rangle = \langle\nabla_\chi\eta, \xi\rangle + \langle\eta, \nabla_\chi\xi\rangle$ *(compatibility with the Riemannian metric),*

for all $\xi, \eta, \chi \in \mathcal{X}(\mathcal{M})$. This affine connection ∇, called the Levi–Civita connection *or the* Riemannian connection *of $\mathcal{M}$ is characterized by the* Koszul formula

$$2\langle\nabla_\chi, \xi\rangle = \chi\langle\eta, \xi\rangle + \eta\langle\xi, \chi\rangle - \xi\langle\chi, \eta\rangle - \langle\chi, [\eta, \xi]\rangle + \langle\eta, [\xi, \chi]\rangle + \langle\xi, [\chi, \eta]\rangle.$$

It is worth noting that the symmetry of the Riemannian connection ensures equality of mixed partial derivatives, and the compatibility with the Riemannian metric ensures that a generalized product rule holds.

When $\mathcal{M}$ is a submanifold of a vector space $\mathcal{E}$, the Riemannian connection reduces to the orthogonal projection of the usual directional derivative. Recall from the previous subsection that, in this case, every vector $\xi_x \in T_x\mathcal{M}$ has a unique decomposition in a tangential component $\mathrm{P}_x\,\xi_x$ and an orthogonal component $\mathrm{P}_x^\perp\,\xi_x$. Then, if ∇ is a Riemannian connection, the covariant derivative of ξ with respect to η reduces to

$$\nabla_{\eta_x}\xi_x = \mathrm{P}_x(\mathrm{D}\,\xi_x[\eta_x]),$$

see [2, Proposition 5.3.2].

Now we are ready to define the Riemannian Hessian operator of a function on $\mathcal{M}$.

Definition 3.14 (Riemannian Hessian)**.** Let f be a real-valued smooth function on a Riemannian manifold $\mathcal{M}$. Then the *Riemannian Hessian* of f at a point x in $\mathcal{M}$ is the linear mapping $\operatorname{Hess} f(x)$ of $T_x\mathcal{M}$ into itself defined by

$$\operatorname{Hess} f(x)[\xi_x] = \nabla_{\xi_x} \operatorname{grad} f(x),$$

for all ξ_x in $T_x\mathcal{M}$, where ∇ is the Riemannain connection on $\mathcal{M}$.

Similarly to Lemma 3.4, we can calculate the Riemannian Hessian on submanifolds from the Hessian in the embedding space.

Lemma 3.15 (Hessians on submanifolds, [2, Section 5.3.3])**.** *Let $\mathcal{M}$ be a Riemannian submanifold of a Euclidean space $\mathcal{E}$. Let $\bar{f} : \mathcal{E} \to \mathbb{R}$ be a function with Euclidean gradient $\operatorname{grad} \bar{f}(x)$ at point $x \in \mathcal{M}$. Then the Riemannian Hessian of $f \coloneqq \bar{f}|_{\mathcal{M}}$ is given by*

$$\operatorname{Hess} f(x)[\xi_x] = \mathrm{P}_x \,\mathrm{D}\left(\mathrm{P}_x \operatorname{grad} \bar{f}(x)\right). \tag{3.7}$$

Using the chain rule, we can write (3.7) as

$$\begin{aligned} \operatorname{Hess} f(x)[\xi] &= \mathrm{P}_x \,\mathrm{D}\left(\mathrm{P}_x \operatorname{grad} \bar{f}(x)\right) \\ &= \mathrm{P}_x \operatorname{Hess} \bar{f}(x)[\xi_x] + \mathrm{P}_x \,\mathrm{D}_\xi \,\mathrm{P}_x \operatorname{grad} \bar{f}(x), \end{aligned} \tag{3.8}$$

where we view $x \mapsto \mathrm{P}_x$ as an operator-valued function and denote its directional derivative by D_ξ. We observe that the first term in (3.8) is just the orthogonal projection of the Euclidean Hessian, while the second one depends on the curvature of the manifold $\mathcal{M}$. Indeed, the second term is equal to zero when $\mathcal{M}$ is flat, i. e. a linear subspace of the embedding Euclidean space, cf. [58, Subsection 4.1]. Clearly, the main challenge in calculating the Riemannian Hessian in (3.8) is the derivative of the projection operator. In [3, Section 3], the authors show the following result.

Lemma 3.16 (derivative of orthogonal projection)**.** *Let $\mathcal{M}$ be a Riemannian submanifold of a Euclidean space $\mathcal{E}$. For any $x \in \mathcal{M}$, let P_x denote the orthogonal projection onto the tangent space $T_x\mathcal{M}$, and $\mathrm{P}_x^\perp \coloneqq \mathrm{id}_\mathcal{E} - \mathrm{P}_x$ the orthogonal projection on its orthogonal complement $(T_x\mathcal{M})^\perp$. We view $x \mapsto \mathrm{P}_x$ as an operator-valued function and denote its Gâteaux derivative at point x in the direction of $\xi \in T_x\mathcal{M}$ by $\mathrm{D}_\xi \,\mathrm{P}_x$. Then*

$$\mathrm{P}_x \,\mathrm{D}_\xi \,\mathrm{P}_x \, u = \mathrm{P}_x \,\mathrm{D}_\xi \,\mathrm{P}_x \left(\mathrm{P}_x^\perp \, u\right), \tag{3.9}$$

for all $x \in \mathcal{M}$, $\xi \in T_x\mathcal{M}$ and $u \in \mathcal{E}$.

By a simple calculation using the symmetry of the Riemannian connection (cf. [2, Propositions 5.5.2 and 5.5.3]), it can be shown that the Riemannian Hessian is self-adjoint with respect to the Riemannian metric, i. e.

$$\langle \operatorname{Hess} f[\xi], \eta \rangle = \langle \xi, \operatorname{Hess} f[\eta] \rangle, \text{ for all } \xi, \eta \in \mathcal{X}(\mathcal{M}).$$

The decisive property of the Riemannain Hessian, which makes second-order optimization methods work, is the fact that, around critical points, it is preserved by a retraction.

Theorem 3.17 (Hessian-retraction property, [2, Proposition 5.5.6])**.** *Let R be a retraction, let x^* be a critical point of a real-valued function f (i. e.* $\operatorname{grad} f(x^*) = 0$*). Then*

$$\operatorname{Hess} f(x^*) = \operatorname{Hess}(f \circ R_{x^*})(0).$$

Algorithm 3.18 Riemannian Newton method on a manifold $\mathcal{M}$, [2, Algorithm 5].

Input: Riemannian manifold $\mathcal{M}$; smooth real-valued function f on $\mathcal{M}$; retraction R; Riemannian connection ∇ on $\mathcal{M}$; initial iterate x_0.

1: **for** $k = 0$ **until** convergence **do**

2: Solve the Newton equation

$$\operatorname{Hess} f(x_k)\eta_k = -\operatorname{grad} f(x_k)$$

for the unknown $\eta_k \in T_{x_k}\mathcal{M}$, where $\operatorname{Hess} f(x_k)\eta_k \coloneqq \nabla_{\eta_k} \operatorname{grad} f(x_k)$

3: Set

$$x_{k+1} \leftarrow R_{x_k}(\eta_k)$$

4: **end for**

Now we can define the Riemannian Newton method in Algorithm 3.18 and quote the main convergence result.

Theorem 3.19 (Newton convergence, [2, Theorem 6.3.2])**.** *Under the requirements and notation of Algorithm 3.18, assume that there exists $x^* \in \mathcal{M}$ such that* $\operatorname{grad} f(x^*) = 0$ *and* $(\operatorname{Hess} f(x^*))^{-1}$ *exists. Then there exists a neighbourhood $\mathcal{U}$ of x^* in $\mathcal{M}$ such that, for all $x_0 \in \mathcal{U}$. Algorithm 3.18 generates an infinite sequence $(x_k)_{k=0}^{\infty}$ converging superlinearly (at least quadratically) to x^*.*

Proof. Similar to the proof in the Euclidean case, using charts. □

3.2.3 The Riemannian Trust-Region Method

In the previous subsection, we have seen that the Newton method in Algorithm 3.18 gives us locally superlinear convergence. However, a plain Newton method has some well-known drawbacks:

i) The convergence radius may be small, i. e. if the initial guess is too far from a critical point the method may diverge.

ii) Each step requires the solution of a linear system. This may be expensive and conceptually difficult if the Hessian operator is not even given explicitly but in terms of the action on a vector in the tangent space, as in (4.7).

There exists a number of strategies for remedying these problems. An intuitive method for globalizing the convergence of a Newton method is to modify the Hessian such that the solution ξ of

$$\operatorname{Hess} f(x_k)[\xi] = -\operatorname{grad} f(x_k) \tag{3.10}$$

defines a descent direction, see [73, Section 3.4] for an overview in the Euclidean case. In [2, Section 6.2] a generalization to the Riemannian case is proposed, replacing the Newton equation with

$$\big(\operatorname{Hess} f(x_k) + E_k\big)[\xi] = -\operatorname{grad} f(x_k),$$

where E_k is a sequence of positive-definite linear operators on the tangent spaces $T_{x_k}\mathcal{M}$.

However, such perturbed Newton methods rely on heuristics, and their general convergence properties are not well understood. Moreover, they still require the solution of a linear system in each iteration. A way to circumvent this are trust-region methods [19], which find a critical point of the function f by minimizing a sequence of constraint quadratic models m_{x_k}.

For a real-valued function f on a Riemannian manifold $\mathcal{M}$, a function m_x is called an *order-q model*, $q > 0$, of $\mathcal{M}$ in $x \in \mathcal{M}$ if there exists a neighbourhood $\mathcal{U}$ of x in $\mathcal{M}$ and a constant $c > 0$ such that

$$\big|f(y) - m_x(y)\big| \leq c\big(\operatorname{dist}(x, y)\big)^{q+1}, \quad \text{for all } y \in \mathcal{U}.$$

It can be shown [2, Proposition 7.1.3] that a model m_x is order-q if and only if there exists a neighbourhood $\mathcal{U}'$ of x in $\mathcal{M}$ and a constant $c' > 0$ such that

$$\big|f(y) - m_x(y)\big| \leq c\big\|R_x^{-1}(y)\big\|^{q+1}, \quad \text{for all } y \in \mathcal{U}.$$

i. e. the order of a model can be assessed using any retraction.

Given a retraction R, this result allows to build a model for f by simply taking a truncated Taylor expansion of

$$\widehat{f}_x \coloneqq f \circ R_x,$$

for any $x \in \mathcal{M}$. The definition of $\widehat{f}_x : T_x\mathcal{M} \to \mathbb{R}$ as a real-valued function on a Euclidean space allows us to use standard results from multivariate analysis. A simple first-order model is then given by

$$\widehat{m}_x = \widehat{f}_x(0_x) + \mathrm{D}\,\widehat{f}_x(0_x)[\xi] = f(x) + \langle \operatorname{grad} f(x), \xi \rangle,$$

where the second equality follows form the rigidity condition of the retraction. A generic second-order model is given by

$$\begin{aligned}\widehat{m}_x &= \widehat{f}_x(0_x) + \mathrm{D}\,\widehat{f}_x(0_x)[\xi] + \tfrac{1}{2}\,\mathrm{D}^2\,\widehat{f}_x(0_x)[\xi, \xi] \\ &= f(x) + \langle \operatorname{grad} f(x), \xi \rangle + \tfrac{1}{2}\big\langle \operatorname{Hess} \widehat{f}(x)[\xi], \xi \big\rangle.\end{aligned}$$

A straightforward and useful modification is obtained by replacing the Euclidean Hessian on the tangent space $\operatorname{Hess} \widehat{f}(x)$ by the Riemannian expression $\operatorname{Hess} f(x)$.

Thus, we can define a model

$$m_x = f(x) + \langle \operatorname{grad} f(x), \xi \rangle + \tfrac{1}{2}\big\langle \operatorname{Hess} f(x)[\xi], \xi \big\rangle,$$

which does not make any use of a retraction. However, Theorem 3.17 only guarantees that m_x matches f up to second order if x is a critical point. In general, we can only

Algorithm 3.20 Riemannian trust-region method on a manifold $\mathcal{M}$, [2, Algorithm 10].

Input: Initial iterate $x_0 \in \mathcal{M}$; parameters $\bar{\Delta} > 0$, $\Delta_0 \in (0, \bar{\Delta})$, $\rho' \in (0, \frac{1}{4})$.

for $k = 0$ **until** convergence **do**
  Obtain η_k by approximately solving (3.11)
  <Test for convergence>
  Evaluate ρ_k from (3.12)
  if $\rho_k < \frac{1}{4}$ **then**
    $\Delta_{k+1} = \frac{1}{4}\Delta_k$
  else if $\rho_k > \frac{3}{4}$ and $\|\eta_k\| = \Delta_k$ **then**
    $\Delta_{k+1} = \min(2\Delta_k, \bar{\Delta})$
  else
    $\Delta_{k+1} = \Delta_k$
  end if
  if $\rho_k > \rho'$ **then**
    $\mathbf{X}_{k+1} = R_{\mathbf{X}_k}(\eta_k)$
  else
    $\mathbf{X}_{k+1} = \mathbf{X}_k$
  end if
end for

prove that it will only give us a first-order model. The model m_x can be shown to be of second order for general x if the retraction R is of second order, i. e. if it preserves second-order information of the exponential map, cf. [2, Proposition 5.5.5]. However, numerical results presented later in the next chapter suggest that, in our case, the result also holds for general points on some tensor manifolds.

The main idea of trust-region methods is solving a model problem

$$\begin{aligned} &\min_{\eta \in T_{x_k}\mathcal{M}} m_{x_k}(\eta) \\ &\text{s. t. } \|\eta\| \leq \Delta_k, \end{aligned} \tag{3.11}$$

for some $\Delta_k \geq 0$ in each iteration k to obtain a search direction η_k. To get meaningful results it is crucial to check how well the model m_{x_k} approximates $\widehat{f}$ in $T_{x_k}\mathcal{M}$ in the neighbourhood of $0_{x_k} \in T_{x_k}\mathcal{M}$. This can be expressed in the form of the quotient

$$\rho_k \coloneqq \frac{\widehat{f}(0_{x_k}) - \widehat{f}(\eta_k)}{m_{x_k}(0_{x_k}) - m_{x_k}(\eta_k)}. \tag{3.12}$$

If ρ_k is small (convergence theory [19, 1] suggests that $\rho' < \frac{1}{4}$ is an appropriate threshold), then the model is very inaccurate: the step must be rejected, and the trust-region radius Δ_k must be reduced. If ρ_k is small but less dramatically so, then the step is accepted but the trust-region radius is reduced. If ρ_k is close to 1, then there is a good agreement between the model and the function over the step, and the trust-region radius can be expanded. If $\rho_k \gg 1$, then the model is inaccurate, but the overall optimization iteration

is producing a significant decrease in the cost. If this is the case and the restriction in (3.11) is active, we can try to expand the trust-region radius as long as we stay below a predefined bound $\bar{\Delta} > 0$. This method is summarized in Algorithm 3.20, cf. [1, Algorithm 1].

The convergence theory follows standard techniques from Euclidean optimization [19]. Under some technical assumptions, it can be shown that Algorithm 3.20 converges globally to a stationary point [1, Theorem 4.4] of f. Locally superlinear convergence to a nondegenerate local minimum can be shown [1, Theorem 4.12] as long as the quadratic term in m_{x_k} is a sufficiently good Hessian approximation of f.

4 Tensor Completion

In this chapter, we consider problems of *tensor completion*, i. e. the problem of finding all entries of some low-rank tensor based on partial information. We will see that Riemannian optimization techniques from Chapter 3 give us a natural framework for dealing with these kinds of problems.

In Section 4.1 we present the problem formulation and give an overview of the state of the art in the literature. In Section 4.2, we cite some basic results about the manifold of low-rank tensors. In Section 4.2, we prove our main result, the Riemannian Hessian on $\mathcal{M}_{\mathbf{r}}$. In Section 4.3, we explain the Riemannian trust-region methods based on exact and approximate Hessian evaluations. In Section 4.4, we present the some numerical experiments for our method on synthetic data, a standard test data set from multilinear statistics, and a test set from hyperspectral imaging.

4.1 Problem Formulation and Literature Overview

We consider least-squares problems of the form

$$\begin{aligned} \min_{\mathbf{X}} f(\mathbf{X}) &= \frac{1}{2}\big\| \mathrm{P}_\Omega \mathbf{X} - \mathrm{P}_\Omega \mathbf{A}\big\|^2 \\ \text{s. t. } \mathbf{X} \in \mathcal{M}_{\mathbf{r}} &\coloneqq \big\{ \mathbf{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d} \;\big|\; \operatorname{rank}_{\mathrm{ML}}(\mathbf{X}) = \mathbf{r} \big\}, \end{aligned} \tag{4.1}$$

where $\operatorname{rank}_{\mathrm{ML}}(\mathbf{X}) \in \mathbb{R}^d$ denotes the multilinear rank of a tensor $\mathbf{X}$, and $\mathrm{P}_\Omega : \mathbb{R}^{n_1 \times \cdots \times n_d} \to \mathbb{R}^{n_1 \times \cdots \times n_d}$ is a linear operator. A typical choice found in the literature is

$$[\mathrm{P}_\Omega \mathbf{X}]_{i_1 \ldots i_d} \coloneqq \begin{cases} x_{i_1 \ldots i_d} & \text{if } (i_1, \ldots, i_d) \in \Omega, \\ 0 & \text{otherwise,} \end{cases}$$

where $\Omega \subset \{1, \ldots, n_1\} \times \cdots \times \{1, \ldots, n_d\}$ denotes the sampling set, i. e. we assume that $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is a tensor whose entries with indices in Ω are known.

The tensor completion problem is a generalization of the matrix completion problem, see, for example, the work by Candès and Recht [16]. Early work on tensor completion has been done by Liu et al. [68], who consider the problem

$$\min_{\mathbf{X}} \|\mathbf{X}\|_* \quad \text{s. t. } \mathrm{P}_\Omega \mathbf{X} = \mathrm{P}_\Omega \mathbf{A} \tag{4.2}$$

in the context of image data recovery, where $\|\cdot\|_*$ is a generalized nuclear norm. Note that (4.2) can be viewed as the dual of (4.1). It ensures convexity for the tensor completion problem at the cost of losing the underlying manifold structure of low-rank tensors. Specifically, it does not give a low-rank solution in the presence of noise, i. e. if $\mathbf{A} \notin \mathcal{M}_{\mathbf{r}}$;

in this case, an additional routine may be needed to truncate the result to low rank. Signoretto et al. [78] and Gandy et al. [26] choose a Tikhonov-like approach by minimizing the a penalized unconstrained function

$$\min_{\mathbf{X}} \frac{1}{2}\big\| \mathrm{P}_\Omega \mathbf{X} - \mathrm{P}_\Omega \mathbf{A}\big\|^2 + \frac{\mu}{2}\|\mathbf{X}\|_*.$$

A Riemannian CG method for (4.1) has been proposed by Kressner et al. [57], which is an extension of Vandereycken's earlier work [91] for the matrix completion problem. The authors show rapid linear convergence of their method with satisfactory reconstruction of missing data for a range of applications. Other Riemannian approaches for matrix completion include the work by Ngo and Saad [72] and Mishra et al. [70], who use a product Graßmann quotient manifold structure. A recent survey on tensor completion methods is given in the preprint by Song et al. [81].

In recent research, second-order methods in Riemannian optimization have generated considerable interest in order to find superlinearly converging methods, see the overview by Absil et al. [2, Chapters 6–8] and the references therein. Boumal and Absil [12] apply these techniques to matrix completion in the Graßmannian framework. Vandereycken [91, Subsection 2.3] derives the Hessian for Riemannian matrix completion with an explicit expression of the singular values. In the higher-order tensor case, Eldén/Savas [24] propose a Newton method for computing a rank-$\mathbf{r}$ tensor approximation, using a Graßmannian approach. Ishteva et al. [45] extend these ideas to construct a Riemannian trust-region scheme. Most recently, Kasai and Mishra [46] proposed a trust-region scheme for tensor completion using a product quotient manifold structure.

In this chapter, we propose a Riemannian trust-region scheme for (4.1) using explicit Tucker decompositions and compare it to a state-of-the-art Riemannian trust-region method in a quotient manifold geometry as used in [46]. We derive the exact expression of the Riemannian Hessian on $\mathcal{M}_\mathbf{r}$ for this manifold geometry by using the Weingarten map proposed by Absil et al. [3]. Our work focuses on the application case of tensor completion and contains tensor approximation as the special case of full sampling, i. e. $|\Omega| = \prod_i n_i$.

4.2 The Geometry of Low-Rank Tensors and Riemannian Optimization on $\mathcal{M}_\mathbf{r}$

As mentioned in the introduction to this chapter, the set of tensors of fixed multilinear rank forms a submanifold in the space of all tensors. We will discuss this in Subsection 4.2.1. Then, in Subsection 4.2.2 we will discuss different retractions, and in Subsection 4.2.3 different vector transports on this manifold. Finally, in Subsection 4.2.4, we will prove the main result of this chapter: the Riemannian Hessian formula on $\mathcal{M}_\mathbf{r}$.

4.2.1 Riemannian Manifold Structure of $\mathcal{M}_\mathbf{r}$

In [82, Theorem 3.6], it is shown that the set $\mathcal{M}_\mathbf{r}$ of tensors of fixed multilinear rank $\mathbf{r} = (r_1, \ldots, r_d)$ forms a smooth embedded submanifold of $\mathbb{R}^{n_1 \times \cdots \times n_d}$. The more general

case, which includes Hilbert spaces, is treated in [88].

By counting the degrees of freedom in (2.10), it follows that

$$\dim(\mathcal{M}_{\mathbf{r}}) = \prod_{i=1}^{d} r_i + \sum_{i=1}^{d} r_i n_i - r_i^2,$$

where the last term accounts for the fact that the Tucker decomposition is invariant to simultaneous transformation of the basis matrix with an invertible matrix and the core tensor with its inverse. Being a submanifold of the Euclidean space $(\mathbb{R}^{n_1 \times \cdots \times n_d}, \langle \cdot, \cdot \rangle)$, the manifold $\mathcal{M}_{\mathbf{r}}$ can be endowed with a Riemannian structure in a natural way with the Frobenius inner product $\langle \cdot, \cdot \rangle$ as the Riemannian metric.

As is proven in [54, Subsection 2.3], the tangent space of $\mathcal{M}_{\mathbf{r}}$ at $\mathbf{X} = \mathbf{C} \bigtimes_{i=1}^{d} U_i$ is parametrized as

$$T_{\mathbf{X}}\mathcal{M}_{\mathbf{r}} = \left\{ \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \dot{U}_i \bigtimes_{j \neq i} U_j \;\middle|\; \dot{\mathbf{C}} \in \mathbb{R}^{r_1 \times \cdots \times r_d},\ \dot{U}_i \in \mathbb{R}^{n_i \times r_i} \text{ with } \dot{U}_i^{\mathrm{T}} U_i = O \right\}, \tag{4.3}$$

and the orthogonal projection $\mathrm{P}_{\mathbf{X}} : \mathbb{R}^{n_1 \times \cdots \times n_d} \to T_{\mathbf{X}}\mathcal{M}_{\mathbf{r}}$ is given by

$$\mathbf{A} \mapsto \left(\mathbf{A} \bigtimes_{j=1}^{d} U_j^{\mathrm{T}} \right) \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{A} \times_i \left(\mathrm{P}_{U_i}^{\perp} \left[\mathbf{A} \bigtimes_{j \neq i} U_j^{\mathrm{T}} \right]_{(i)} C_{(i)}^{+} \right) \bigtimes_{k \neq i} U_k, \tag{4.4}$$

where $C_{(i)}^{+}$ denotes the Moore-Penrose pseudoinverse of $C_{(i)}$. Note that $C_{(i)}$ has full row rank, i. e. $C_{(i)}^{+} = C_{(i)}^{\mathrm{T}} (C_{(i)} C_{(i)}^{\mathrm{T}})^{-1}$. We use $\mathrm{P}_{U_i}^{\perp} = I_{n_i} - U_i U_i^{\mathrm{T}}$ to denote the orthogonal projection onto $\operatorname{span}(U_i)^{\perp}$.

Furthermore, it can be shown that the HOSVD (2.20) is locally a C^{∞} function in the manifold topology of $\mathcal{M}_{\mathbf{r}}$, see [57, Proposition 2.1] for further details. This allows us its use in continuous optimization, as we will see in the next section.

4.2.2 Retraction on $\mathcal{M}_{\mathbf{r}}$

In this subsection, we will introduce retractions on the manifold $\mathcal{M}_{\mathbf{r}}$ (see Definition 3.5). As has been shown in in [57], the truncated HOSVD (2.20) defined in Subsection 2.2.2 satisfies these conditions.

Lemma 4.1 (HOSVD as a retraction, [57, Proposition 2.3])**.** *The map*

$$R : T\mathcal{M}_{\mathbf{r}} \to \mathcal{M}_{\mathbf{r}},\ (\mathbf{X}, \xi) \mapsto \mathrm{P}_{\mathbf{r}}^{\mathrm{HO}}(\mathbf{X} + \xi)$$

is a retraction on $\mathcal{M}_{\mathbf{r}}$*.*

The above lemma makes use of the fact that $\mathcal{M}_{\mathbf{r}}$ is a submanifold of $\mathbb{R}^{n_1 \times \cdots n_d}$. However, it can also be viewed as a quotient manifold, where the quotient structure comes from the indeterminacy in the Tucker decomposition. More specifically,

$$\mathcal{M}_{\mathbf{r}} = \big(\mathbb{R}_{*}^{r_1 \times \cdots \times r_d} \times \mathrm{St}(r_1, n_1) \times \cdots \times \mathrm{St}(r_d, n_d) \big) \Big/ \big((\mathrm{O}(r_1) \times \cdots \times \mathrm{O}(r_d) \big),$$

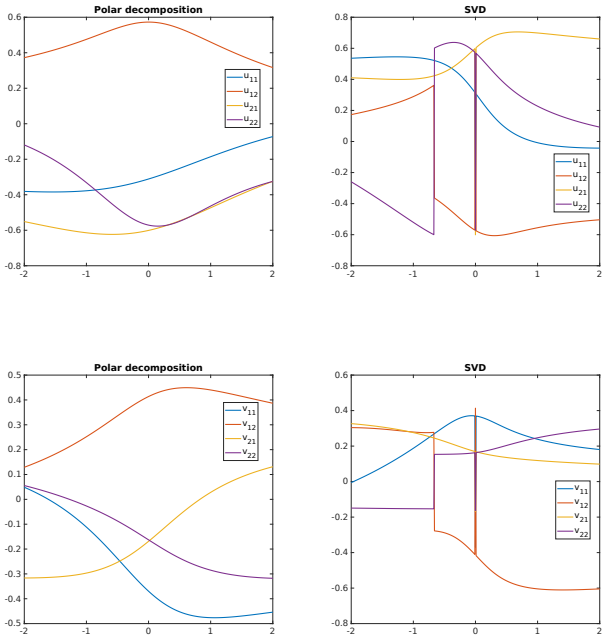


Figure 4.1: Retraction by HOSVD and polar decomposition in $d = 2$. With $\mathbf{X} \in \mathcal{M}_\mathbf{r}$, $\xi \in T_\mathbf{X}\mathcal{M}_\mathbf{r}$, retraction $t \mapsto R_\mathbf{X}(t\xi)$ is computed for $t \in [-2, 2]$. Two first components of U_1 and U_2 are plotted.

with

$$\mathbb{R}_*^{r_1 \times \cdots \times r_d} = \left\{ \mathbf{C} \in \mathbb{R}^{r_1 \times \cdots \times r_d}, \;\middle|\; \operatorname{rank}_{\mathrm{ML}}(\mathbf{C}) = \mathbf{r} \right\},$$

the manifold of full-rank tensors,

$$\operatorname{St}(r_i, n_i) = \left\{ U_i \in \mathbb{R}^{n_i \times r_i} \,\middle|\, U_i^{\mathrm{T}} U_i = I_{r_i} \right\},$$

the *Stiefel manifold*, and

$$\mathrm{O}(r_i) = \left\{ Q_i \in \mathbb{R}^{r_i \times r_i} \,\middle|\, Q_i^{\mathrm{T}} Q_i = I_{r_i} \right\},$$

the orthogonal group.

Using this property, it can be shown that the orthonormal factors of the *polar decomposition* for the basis matrix variations also give us a retraction, cf. [46].

Lemma 4.2 (Polar decomposition as a retraction)**.** *Let*

$$\mathbf{X} = \mathbf{C} \bigtimes_{i=1}^{d} U_i \in \mathcal{M}_\mathbf{r},$$

and

$$\dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \dot{U}_i \bigtimes_{j \neq i} U_j \in T\mathcal{M}_\mathbf{r}.$$

Then the map

$$R : T\mathcal{M}_{\mathbf{r}} \to \mathcal{M}_{\mathbf{r}},\ (\mathbf{X}, \xi) \mapsto \mathbf{D} \bigtimes_{i=1}^{d} V_i,$$

defined by

$$\begin{aligned}
&\mathbf{D} = \mathbf{C} + \dot{\mathbf{C}},\\
&V_i = (U_i + \dot{U}_i)\big((U_i + \dot{U}_i)^{\mathrm{T}}(U_i + \dot{U}_i)\big),\ i = 1, \ldots, d,
\end{aligned}$$

is a retraction on $\mathcal{M}_{\mathbf{r}}$.

Both retractions allow their use for optimization methods on $\mathcal{M}_{\mathbf{r}}$. However, there is an important difference: while both are smooth around a point $\mathcal{M}_{\mathbf{r}}$ on the manifold, the "smoothness radius" of the retraction by HOSVD tends to be much smaller, see Figure 4.1. We will see the significance of this in the following subsection.

4.2.3 Vector Transport on $\mathcal{M}_{\mathbf{r}}$

A simple vector transport associated with a retraction R is given by the orthogonal projection onto the tangent space, i. e. $\mathcal{T}_\eta(\xi) = \mathrm{P}_{R_{\mathbf{X}}(\eta)}(\xi)$, see [2, Subsection 8.1.3]; in our case, this is the formula (4.4).

In recent literature, there has been particular interest towards *intrinsic representations* of tangent vectors and associated vector transports, cf. [43]. These representations are based on the idea that the tangent space of a p-dimensional manifold is a p-dimensional vector space, and thus, all tangent vector admit a basis representation by coordinate vector $v \in \mathbb{R}^p$. This makes tangent vector operations easy since everything can be done in $\mathbb{R}^p$. However, the computation of the basis representation is non-trivial and poses computational and theoretical challenges. Here, we present a novel approach to the intrinsic representation of $T_{\mathbf{X}}\mathcal{M}_{\mathbf{r}}$ based on the matrix case treated in [43].

For a p-dimensional manifold $\mathcal{M}$ of w-dimensional vector space $\mathcal{E}$, we define, on an open subset $\mathcal{U} \in \mathcal{M}$, a smooth tangent basis field

$$B : \mathcal{U} \to \mathbb{R}^{w \times p},\ x \mapsto B_x.$$

We see the application of B_x to $v \in \mathbb{R}^p$ as a matrix-vector-multiplication which returns the element $\xi \in T_x\mathcal{M}$ for which v is the basis coefficient vector. The inverse operation B_x^+ applied to $\xi \in T_x\mathcal{M}$ returns the basis coefficients. It can be shown that

$$\mathcal{T}_\eta \xi = B_{R_x(\eta)} B_x^+ \xi$$

defines a vector transport on $\mathcal{M}$.

Now we consider the maps converting the intrinsic representation of a tangent vector to the extrinsic one, and vice versa. Let

$$\mathrm{D2E}_x^{\mathcal{M}} : v \mapsto \xi = B_x v, \text{ and } \mathrm{E2D}_x^{\mathcal{M}} : \xi \mapsto B_x^+ \eta.$$

Then

$$\mathrm{E2D}^{\mathcal{M}}_{R_x(\eta)} \circ \mathcal{T}_\eta \xi \circ \mathrm{D2E}^{\mathcal{M}}_x v = B_y^+(B_y B_x^+(B_x v)) = v,$$

i. e. in the intrinsic representation, the vector transport is just the identity.

Now, we present the application of this idea to $\mathcal{M}_{\mathbf{r}}$. Note that, in order to apply a vector transport in intrinsic coordinates, we need a locally smoothly varying basis field. Since the retraction by HOSVD may introduce non-smoothness even for small perturbations (see the previous subsection), it does not lend itself to this representation. We recall the tangent space of $\mathcal{M}_{\mathbf{r}}$ given by

$$T_{\mathbf{X}}\mathcal{M}_{\mathbf{r}} = \left\{ \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \dot{U}_i \bigtimes_{j\neq i} U_j \;\middle|\; \dot{\mathbf{C}} \in \mathbb{R}^{r_1\times\cdots\times r_d},\ \dot{U}_i \in \mathbb{R}^{n_i\times r_i} \text{ with } \dot{U}_i^{\mathrm{T}} U_i = O \right\}$$

Now let $U_{i\perp} \in \mathbb{R}^{n_i\times(n_i-r_i)}$ a matrix whose columns form an orthonormal basis for $\operatorname{span}(U_i)^\perp$, i. e. $\begin{bmatrix} U_i & U_{i\perp}\end{bmatrix}^{\mathrm{T}}\begin{bmatrix} U_i & U_{i\perp}\end{bmatrix} = I_{n_i}$. Then, because of the orthogonality conditions, a matrix $K_i \in \mathbb{R}^{(n_i-r_i)\times r_i}$ exists such that $U_{i\perp}K_i = \dot{U}_i$. Moreover, since $U_{i\perp}$ has full rank, K_i is uniquely determined by $U_{i\perp}^+\dot{U}_i$. Hence $T_{\mathbf{X}}\mathcal{M}_{\mathbf{r}}$ can be written as

$$\begin{aligned}
T_{\mathbf{X}}\mathcal{M}_{\mathbf{r}} &= \left\{ \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \left(U_{i\perp}K_i\right) \bigtimes_{j\neq i} U_j \;\middle|\; \dot{\mathbf{C}} \in \mathbb{R}^{r_1\times\cdots\times r_d},\ K_i \in \mathbb{R}^{(n_i-r_i)\times r_i} \right\} \\
&= \left\{ \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \left[K_i C_{(i)}\right]^{(i)} \times_i U_{i\perp} \bigtimes_{j\neq i} U_j \;\middle|\; \dot{\mathbf{C}} \in \mathbb{R}^{r_1\times\cdots\times r_d},\ K_i \in \mathbb{R}^{(n_i-r_i)\times r_i} \right\} \\
&= \left\{ \mathbf{G} \bigtimes_{i=1}^{d} \begin{bmatrix} U_i & U_{i\perp}\end{bmatrix} \;\middle|\; \dot{\mathbf{C}} \in \mathbb{R}^{r_1\times\cdots\times r_d},\ K_i \in \mathbb{R}^{(n_i-r_i)\times r_i} \right\},
\end{aligned}$$

the tensor $\mathbf{G} = \mathbf{G}(\mathbf{C}, \dot{\mathbf{C}}, K_1, \ldots, K_d) \in \mathbb{R}^{n_1\times\cdots\times n_d}$ being given by

$$g_{\nu_1\cdots\nu_d} = \begin{cases} \dot{c}_{\nu_1\cdots\nu_d}, & \text{if } \ \nu_i \le r_i \ \text{ for all } i, \\ \sum_{\mu=1}^{r_i} k^{(i)}_{(\nu_i-r_i),\mu} c_{\nu_1\cdots\nu_{i-1}\mu\nu_{i+1}\cdots\nu_d}, & \text{if } \ \nu_i > r_i \ \text{ and } \nu_j \le r_j \ \text{ for all } i\neq j, \\ 0, & \text{else,} \end{cases}$$

where $k^{(i)}_{\nu,\mu}$ denotes the entry of K_i in the νth row and μth column. Figure 4.2 illustrates this structure for $d = 3$.

Recall that the Riemannian metric is given by the Frobenius inner product, and using the orthogonality relations we obtain

$$\langle \xi, \xi' \rangle = \langle \dot{\mathbf{C}}, \dot{\mathbf{C}}' \rangle + \sum_{i=1}^{d} \left\langle \mathbf{C}, \mathbf{C} \times_i \dot{U}_i^{\mathrm{T}} \dot{U}_i' \right\rangle = \langle \dot{\mathbf{C}}, \dot{\mathbf{C}}' \rangle + \sum_{i=1}^{d} \left\langle \mathbf{C}, \mathbf{C} \times_i K_i^{\mathrm{T}} K_i' \right\rangle.$$

With the notations

$$\mathcal{C} \coloneqq \left\{ \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i \;\middle|\; \dot{\mathbf{C}} \in \mathbb{R}^{r_1\times\cdots\times r_d} \right\}, \quad \mathcal{U}_i \coloneqq \left\{ \mathbf{C} \times_i \left(U_{i\perp}K_i\right) \bigtimes_{j\neq i} U_j \;\middle|\; K_i \in \mathbb{R}^{(n_i-r_i)\times r_i} \right\},$$

$\mathbf{G}(\mathbf{C}, \dot{\mathbf{C}}, K_1, \ldots, K_d) =$

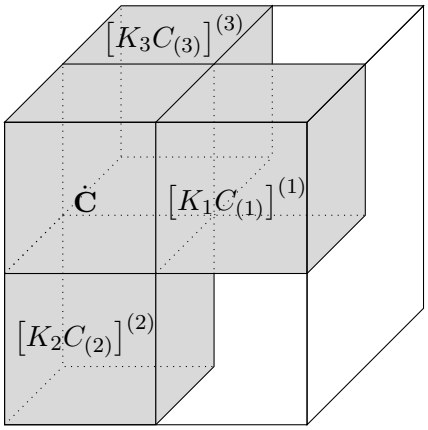


Figure 4.2: Structure of the coefficient tensor $\mathbf{G}$ for $d = 3$.

the tangent space at $\mathbf{X} \in \mathcal{M}_\mathbf{r}$ can be written as the direct sum

$$T_\mathbf{X}\mathcal{M}_\mathbf{r} = \mathcal{C} \oplus \mathcal{U}_1 \oplus \mathcal{U}_2 \oplus \cdots \oplus \mathcal{U}_d,$$

and the summands are pairwise orthogonal spaces.

Let $\mathbf{e}_i^{(n)}$ denote the ith coordinate vector of $\mathbb{R}^n$. Then, the canonical basis of $\mathcal{C}$ is simply given by

$$\left\{ \left(\mathbf{e}_{\nu_1}^{(r_1)} \circ \cdots \circ \mathbf{e}_{\nu_d}^{(r_d)} \right) \bigtimes_{i=1}^{d} U_i \;\middle|\; 1 \le \nu_i \le r_i \text{ for all } i \right\},$$

and the canonical basis for a space $\mathcal{U}_i$ is given by

$$\left\{ \mathbf{C} \times_i \left(U_{i\perp} \left(\mathbf{e}_{\nu_i}^{(n_i)} \circ \mathbf{e}_{\mu_i}^{(r_i)} \right) \Sigma_i^{-1} \right) \bigtimes_{j \ne i} U_j \;\middle|\; 1 \le \nu_i \le n_i \text{ and } 1 \le \mu_i \le r_i \right\},$$

where $\Sigma_i = C_{(i)} C_{(i)}^{\mathrm{T}}$.

Now we can compute the functions $\mathrm{E2D}_\mathbf{X}^{\mathcal{M}_\mathbf{r}}$ and $\mathrm{D2E}_\mathbf{X}^{\mathcal{M}_\mathbf{r}}$ in Algorithms 4.3 and 4.4.

The functions α_{U_i} from Algorithm 4.3 and β_{U_i} from Algorithm 4.4 can be computed efficiently using Householder reflectors, see [43].

4.2.4 The Riemannian Hessian on $\mathcal{M}_\mathbf{r}$

By Lemma 3.4, the Riemannian gradient of the tensor completion cost function is given by

$$\operatorname{grad} f(\mathbf{X}) = \mathrm{P}_\mathbf{X}(\mathrm{P}_\Omega \mathbf{X} - \mathrm{P}_\Omega \mathbf{A}). \tag{4.5}$$

Using the sparsity of $\mathrm{P}_\Omega \mathbf{X} - \mathrm{P}_\Omega \mathbf{A}$, a gradient evaluation requires $\mathcal{O}(r^d(|\Omega| + n) + r^{d+1})$ operations, cf. [57, Subsection 3.1], where we assume that the r_i and n_i are constant in each mode for simplicity of notation.

Algorithm 4.3 Compute $\mathrm{E2D}_{\mathbf{X}}^{\mathcal{M}_\mathbf{r}}(\xi)$.

Input: $\mathbf{X} = \mathbf{C} \bigtimes_{i=1}^{d} U_i \in \mathcal{M}_\mathbf{r}$, $\xi_\mathbf{X} = \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \dot{U}_i \bigtimes_{j \neq i} U_j \in T_\mathbf{X}\mathcal{M}_\mathbf{r}$, represented by $(\dot{\mathbf{C}}, \dot{U}_1, \dots, \dot{U}_d)$, and functions $\alpha_{U_i} : \mathbb{R}^{n_i \times r_i} \to \mathbb{R}^{n_i \times r_i}$, $A \mapsto [\, U_i \; U_{i\perp} \,]^{\mathrm{T}} A$, for all i.

1: **for** $i = 1, \dots, d$ **do**
2: $\quad K_i = (\alpha_{U_i}(\dot{U}_i \Sigma_i))_{(r_i+1:n_i,:)}$, where $M_{(a:b,:)}$ denotes the submatrix formed by ath to bth rows of the matrix M
3: **end for**
4: Reshape $\dot{\mathbf{C}}, K_1, \dots, K_d$ to be column vectors; stack them to make $\mathbf{v}_\mathbf{X} \in \mathbb{R}^{\dim(\mathcal{M}_\mathbf{r})}$
5: **return** vector $\mathbf{v}_\mathbf{X}$

Algorithm 4.4 Compute $\mathrm{D2E}_{\mathbf{X}}^{\mathcal{M}_\mathbf{r}}(\mathbf{v}_\mathbf{X})$.

Input: $\mathbf{X} = \mathbf{C} \bigtimes_{i=1}^{d} U_i \in \mathcal{M}_\mathbf{r}$, $\mathbf{v}_\mathbf{X} \in \mathbb{R}^{\dim(\mathcal{M}_\mathbf{r})}$, and functions $\beta_{U_i} : \mathbb{R}^{n_i \times r_i} \to \mathbb{R}^{n_i \times r_i}$, $A \mapsto [\, U_i \; U_{i\perp} \,] A$, for all i.

1: Reshape the first $\prod_i r_i$ entries of $\mathbf{v}_\mathbf{X}$ to be $\dot{\mathbf{C}} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$
2: **for** $i = 1, \dots, d$ **do**
3: $\quad$ Reshape the next $(n_i - r_i) r_i$ entries of $\mathbf{v}_\mathbf{X}$ to be $K_i \in \mathbb{R}^{(n_i - r_i) \times r_i}$
4: $\quad$ Compute $\dot{U}_i = \beta_{U_i}\Big(\begin{bmatrix} O \\ K_i \Sigma_i^{-1} \end{bmatrix} \Big)$
5: **end for**
6: **return** $(\dot{\mathbf{C}}, \dot{U}_1, \dots, \dot{U}_d)$, which represents $\xi_\mathbf{X} = \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \dot{U}_i \bigtimes_{j \neq i} U_j$

Now we would like to compute the Riemannian Hessian of the tensor completion cost function. A finite-difference approximation can be defined in different ways. An intuitive formula is given by

$$H^{\mathrm{FD}}[\xi] = \frac{\mathcal{T}_\xi \operatorname{grad} f(R_\mathbf{X}(h\xi)) - \operatorname{grad} f(\mathbf{X})}{h}, \tag{4.6}$$

see, for example, [2, Subsection 8.2.1]. However, such a mapping will in general not be linear [11], and should be applied with care, as theoretical understanding is yet incomplete.

The Lemma 3.16 can be applied to the case of the low-rank Tucker manifold $\mathcal{M} = \mathcal{M}_\mathbf{r}$. First, we calculate the derivative $\mathrm{D}_\xi \mathrm{P}_\mathbf{X}$.

Lemma 4.5 (derivative of $\mathrm{P}_\mathbf{X}$)**.** *Let $\mathbf{X} \in \mathcal{M}_\mathbf{r}$ be a tensor on the low-rank manifold, given by the factorization $\mathbf{X} = \mathbf{C} \bigtimes_{i=1}^{d} U_i$, and let $\xi \in T_\mathbf{X}\mathcal{M}_\mathbf{r}$, given by the variations*

$$\xi = \dot{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \dot{U}_i \bigtimes_{j \neq i} U_j.$$

We use the notations $\mathrm{P}_{U_i} = U_i U_i^{\mathrm{T}}$, $\mathrm{P}_{U_i}^{\perp} = I_{n_i} - U_i U_i^{\mathrm{T}}$ and $\dot{\mathrm{P}}_{U_i} = \dot{U}_i U_i^{\mathrm{T}} + U_i \dot{U}_i^{\mathrm{T}}$. Then,

for any $\mathbf{E} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$*, the derivative of* $\mathrm{P}_{\mathbf{X}}$ *in the direction of* ξ *is given by*

$$\begin{aligned}
\mathrm{D}_{\xi} \mathrm{P}_{\mathbf{X}} \mathbf{E} = \sum_{i=1}^{d} \Bigg\{ & \mathbf{E} \times_i \dot{\mathrm{P}}_{U_i} \bigtimes_{j \neq i} \mathrm{P}_{U_j} \\
& + \dot{\mathbf{C}} \times_i \left(\mathrm{P}_{U_i}^{\perp} \left[\mathbf{E} \bigtimes_{j \neq i} U_j^{\mathrm{T}} \right]_{(i)} C_{(i)}^{+} \right) \bigtimes_{k \neq i} U_k \\
& - \mathbf{C} \times_i \left(\dot{\mathrm{P}}_{U_i} \left[\mathbf{E} \bigtimes_{j \neq i} U_j^{\mathrm{T}} \right]_{(i)} C_{(i)}^{+} \right) \bigtimes_{k \neq i} U_k \\
& + \sum_{l \neq i} \mathbf{C} \times_i \left(\mathrm{P}_{U_i}^{\perp} \left[\mathbf{E} \times_l \dot{U}_l^{\mathrm{T}} \bigtimes_{l \neq j \neq i} U_j^{\mathrm{T}} \right]_{(i)} C_{(i)}^{+} \right) \bigtimes_{k \neq i} U_k \\
& + \mathbf{C} \times_i \left(\mathrm{P}_{U_i}^{\perp} \left[\mathbf{E} \bigtimes_{j \neq i} U_j^{\mathrm{T}} \right]_{(i)} \left[\left(I - C_{(i)}^{+} C_{(i)} \right) \dot{C}_{(i)}^{\mathrm{T}} C_{(i)}^{+\mathrm{T}} C_{(i)}^{+} - C_{(i)}^{+} \dot{C}_{(i)} C_{(i)}^{+} \right] \right) \bigtimes_{k \neq i} U_k \\
& + \sum_{l \neq i} \mathbf{C} \times_i \left(\mathrm{P}_{U_i}^{\perp} \left[\mathbf{E} \bigtimes_{j \neq i} U_j^{\mathrm{T}} \right]_{(i)} C_{(i)}^{+} \right) \times_l \dot{U}_l \bigtimes_{l \neq k \neq i} U_j \Bigg\},
\end{aligned}$$

where $I = I_{\prod_{j \neq i} r_j}$ *is the identity matrix of the appropriate size.*

Proof. The formula can be obtained by identifying the tensor $\mathbf{X}$ with the factors in the Tucker decomposition and viewing the orthogonal projection defined in (4.4) as a function

$$\mathrm{P}_{\cdot}\, \mathbf{E} : \mathbb{R}^{r_1 \times \cdots \times r_d} \times \mathbb{R}^{n_1 \times r_1} \times \cdots \times \mathbb{R}^{n_d \times r_d} \to \mathbb{R}^{n_1 \times \cdots \times n_d},\ (\mathbf{C}, U_1, \ldots, U_d) \mapsto \mathrm{P}_{\mathbf{X}} \mathbf{E},$$

for any $\mathbf{E} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. For calculating the derivative of the pseudoinverse, we use the formula given in [29, Theorem 4.3], i. e.

$$\mathrm{D}_{\dot{C}} \left(C^{+} \right) = \left(I - C^{+} C \right) \dot{C}^{\mathrm{T}} C^{+\mathrm{T}} C^{+} + C^{+} C^{+\mathrm{T}} \dot{C}^{\mathrm{T}} \left(C C^{+} - I \right) - C^{+} \dot{C} C^{+},$$

and note that, here, the second term vanishes since $C = C_{(i)}$ has full row rank, and thus the pseudoinverse is a right inverse. □

Using this result, we can immediately evaluate the curvature term in (3.8).

Corollary 4.6 (curvature term)**.** *We use the setting of Lemma 4.5 and denote the orthogonal projection onto* $(T_{\mathbf{X}} \mathcal{M}_{\mathbf{r}})^{\perp}$ *by* $\mathrm{P}_{\mathbf{X}}^{\perp} \coloneqq \mathrm{id} - \mathrm{P}_{\mathbf{X}}$*. Then*

$$\mathrm{P}_{\mathbf{X}} \mathrm{D}_{\xi} \mathrm{P}_{\mathbf{X}} \mathrm{P}_{\mathbf{X}}^{\perp} \mathbf{E} = \widetilde{\mathbf{C}} \bigtimes_{i=1}^{d} U_i + \sum_{i=1}^{d} \mathbf{C} \times_i \widetilde{U}_i \bigtimes_{j \neq i} U_j \in T_{\mathbf{X}} \mathcal{M}_{\mathbf{r}},$$

with

$$\begin{aligned}
\widetilde{\mathbf{C}} &= \sum_{j=1}^{d} \left(\mathbf{E} \times_j \dot{U}_j^{\mathrm{T}} \bigtimes_{k \neq j} U_k^{\mathrm{T}} - \mathbf{C} \times_j \left(\dot{U}_j^{\mathrm{T}} \left[\mathbf{E} \bigtimes_{k \neq j} U_k^{\mathrm{T}} \right]_{(j)} C_{(j)}^{+} \right) \right), \\
\widetilde{U}_i &= \mathrm{P}_{U_i}^{\perp} \left(\left[\mathbf{E} \bigtimes_{j \neq i} U_j^{\mathrm{T}} \right]_{(i)} \left(I - C_{(i)}^{+} C_{(i)} \right) \dot{C}_{(i)}^{\mathrm{T}} C_{(i)}^{+\mathrm{T}} + \sum_{k \neq i} \left[\mathbf{E} \times_k \dot{U}_k^{\mathrm{T}} \bigtimes_{k \neq j \neq i} U_j^{\mathrm{T}} \right]_{(i)} \right) C_{(i)}^{+},
\end{aligned}$$

Proof. The result follows by applying Lemma 4.5 to $\mathrm{P}_{\mathbf{X}}^{\perp}\,\mathbf{E} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ after some lengthy but straightforward calculations, using the orthonormality relations $\dot{U}_i^{\mathrm{T}} U_i = O$, $U_i^{\mathrm{T}} U_i = I$ and the rules (2.8) and (2.9) for the matrix-tensor product. □

Thus, with Lemma 3.16, the Riemannian Hessian of the function $f : \mathcal{M}_{\mathbf{r}} \to \mathbb{R}$,

$$f(\mathbf{X}) = \frac{1}{2}\big\| \mathrm{P}_{\Omega}\,\mathbf{X} - \mathrm{P}_{\Omega}\,\mathbf{A}\big\|^2,$$

can be written as

$$\operatorname{Hess} f(\mathbf{X})[\xi] = \mathrm{P}_{\Omega}(\xi) + \mathrm{P}_{\mathbf{X}}\,\mathrm{D}_{\xi}\,\mathrm{P}_{\mathbf{X}}\,\mathrm{P}_{\mathbf{X}}^{\perp}(\mathrm{P}_{\Omega}\,\mathbf{X} - \mathrm{P}_{\Omega}\,\mathbf{A}) \tag{4.7}$$

and the second term can be evaluated with Corollary 4.6.

Note that for an efficient computation of the terms $\widetilde{U}_i$, it is advantageous to multiply out the term containing $I - C_{(i)}^{+} C_{(i)}$. Then, the computation of $\operatorname{Hess} f(\mathbf{X})[\xi]$ for any given $\xi \in T_{\mathbf{X}}\mathcal{M}_{\mathbf{r}}$ has the same complexity as the computation of the gradient, i. e. $\mathcal{O}(r^d(|\Omega| + n) + r^{d+1})$.

Remark 4.7. For the matrix case $d = 2$, the Hessian expression (4.7) can be simplified to recover the expression shown in [91, 3],

$$\begin{aligned}\operatorname{Hess} f(X)[\xi] = {} & \mathrm{P}_U\,\mathrm{P}_{\Omega}(\xi)\,\mathrm{P}_V + \mathrm{P}_U^{\perp}\left[\mathrm{P}_{\Omega}(\xi) + \mathrm{P}_{\Omega}(X - A)\dot{V}\Sigma^{-1}V^{\mathrm{T}}\right]\mathrm{P}_V \\ & + \mathrm{P}_U\left[\mathrm{P}_{\Omega}(\xi) + U\Sigma^{-1}\dot{U}^{\mathrm{T}}\,\mathrm{P}_{\Omega}(X - A)\right]\mathrm{P}_V^{\perp},\end{aligned}$$

where we identify the Tucker decomposition with the usual notation for the SVD, i. e. $U = U_1$, $V = U_2$ and $\Sigma = C$.

4.3 Riemannian Models on $\mathcal{M}_{\mathbf{r}}$

We consider the manifold $\mathcal{M}_{\mathbf{r}}$ of fixed-rank tensors and would like to assess the quality of different model functions. In accordance with Subsection 3.2.3, we consider a first-order model

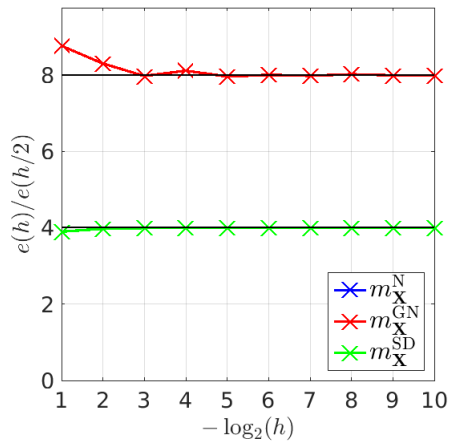
$$m_{\mathbf{X}}^{\mathrm{SD}}(\xi) \coloneqq f(\mathbf{X}) + \langle \operatorname{grad} f(\mathbf{X}), \xi\rangle + \tfrac{1}{2}\langle \xi, \xi\rangle, \tag{4.8}$$

where the superscript indicates that this model corresponds to a steepest-descent method, and a second-order model

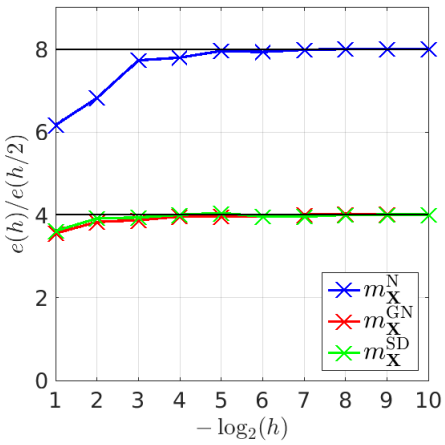
$$m_{\mathbf{X}}^{\mathrm{N}}(\xi) \coloneqq f(\mathbf{X}) + \langle \operatorname{grad} f(\mathbf{X}), \xi\rangle + \tfrac{1}{2}\big\langle \operatorname{Hess} f(\mathbf{X})[\xi], \xi\big\rangle,$$

where the superscript indicates that this model corresponds to a Newton method. Furthermore, we would like to assess the quality of a Hessian approximation which drops the curvature term in Corollary 4.6 and thus ignores the second-order geometry of $\mathcal{M}_{\mathbf{r}}$. This is given by omitting the second term in (4.7), and just considering the projection of the Euclidean Hessian i. e.

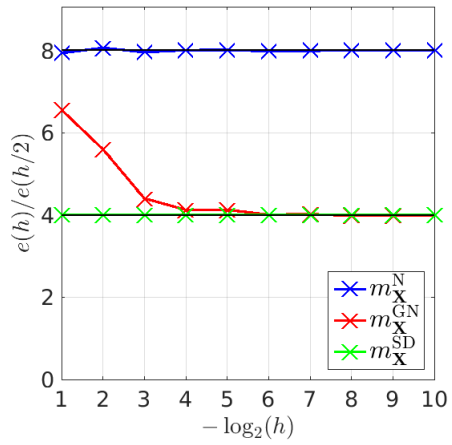
$$\widetilde{\operatorname{Hess}} f(\mathbf{X})[\xi] = \mathrm{P}_{\Omega}\,\xi. \tag{4.9}$$



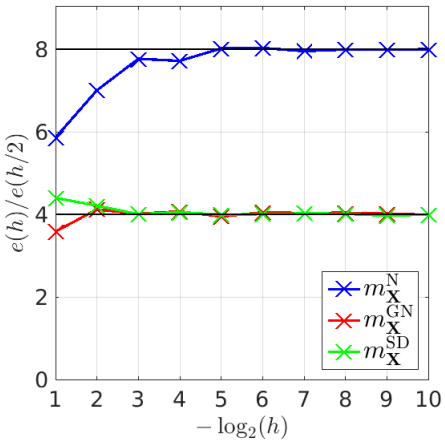
$f(\mathbf{X}) = 0$



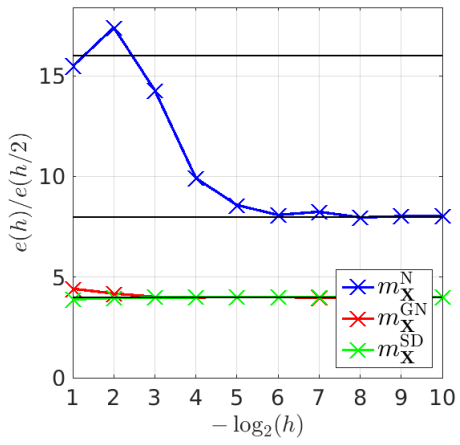
$f(\mathbf{X}) = 1.2 \times 10$



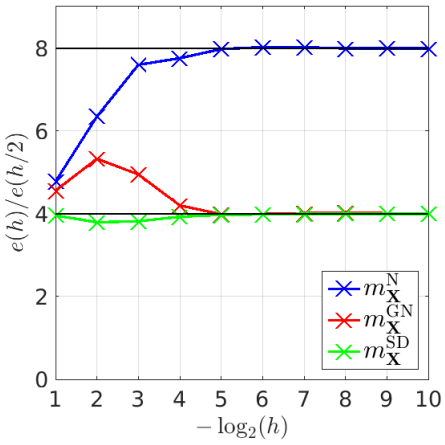
$f(\mathbf{X}) = 2.1 \times 10^{-2}$



$f(\mathbf{X}) = 1.9 \times 10^{2}$

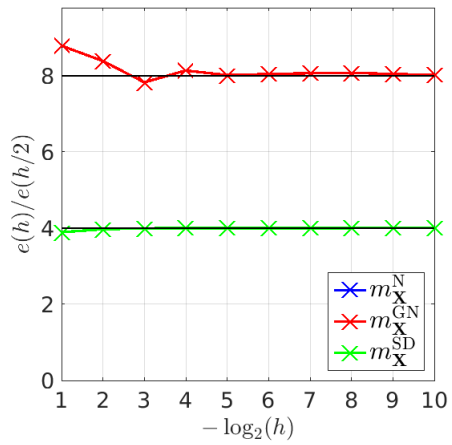


$f(\mathbf{X}) = 3.6 \times 10^{2}$

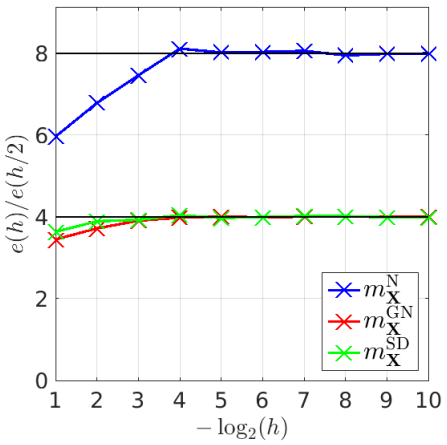


$f(\mathbf{X}) = 1.8 \times 10^{3}$

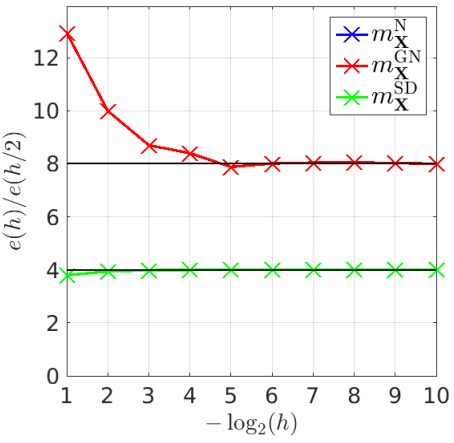
Figure 4.3: The unknown tensor $\mathbf{A}$ has full rank, i. e. $\mathbf{A} \notin \mathcal{M}_{\mathbf{r}}$.



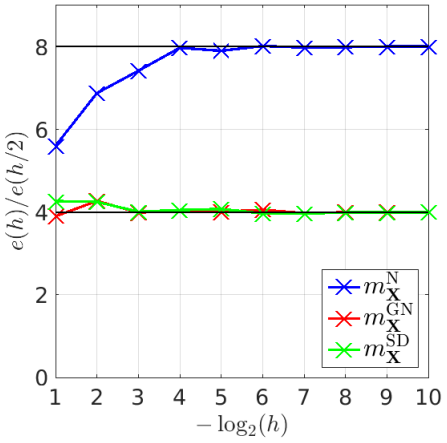
$f(\mathbf{X}) = 0$



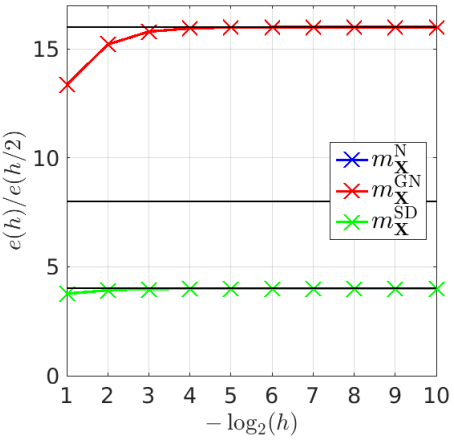
$f(\mathbf{X}) = 1.3 \times 10$



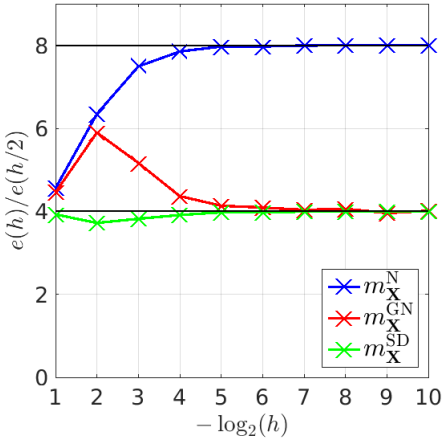
$f(\mathbf{X}) = 0$



$f(\mathbf{X}) = 1.5 \times 10^2$



$f(\mathbf{X}) = 0$



$f(\mathbf{X}) = 1.4 \times 10^3$

Figure 4.4: The unknown tensor $\mathbf{A}$ has low rank, i. e. $\mathbf{A} \in \mathcal{M}_{\mathbf{r}}$.

Omitting the curvature term of the Hessian corresponds to a Riemannian Gauß–Newton method, as described in [2, Subsection 8.4.1]. Thus, we consider the model function

$$m_{\mathbf{X}}^{\mathrm{GN}}(\xi) \coloneqq f(\mathbf{X}) + \langle \operatorname{grad} f(\mathbf{X}), \xi \rangle + \tfrac{1}{2} \big\langle \widetilde{\operatorname{Hess}} f(\mathbf{X})[\xi], \xi \big\rangle. \tag{4.10}$$

As usual, we can expect a Gauß–Newton method to converge superlinearly (as the corresponding model to be of order higher than 1) if the residual of the least-squares problem is low. This can be seen in terms of (4.7), where the curvature term is given as

$$\big(\operatorname{Hess} f(\mathbf{X}) - \widetilde{\operatorname{Hess}} f(\mathbf{X}) \big)[\xi] = \mathrm{P}_{\mathbf{X}} \, \mathrm{D}_{\xi} \, \mathrm{P}_{\mathbf{X}} \, \mathrm{P}_{\mathbf{X}}^{\perp} (\mathrm{P}_{\Omega} \, \mathbf{X} - \mathrm{P}_{\Omega} \, \mathbf{A}),$$

which is clearly equal to zero if $\mathrm{P}_{\Omega} \, \mathbf{X} = \mathrm{P}_{\Omega} \, \mathbf{A}$ and hence $f(\mathbf{X}) = 0$.

To assess the order of a model, we define for a given $m_{\mathbf{X}}$ the *model error*

$$e(\xi, h) \coloneqq \big| \widehat{f}_{\mathbf{X}}(h\xi) - m_{\mathbf{X}}(h\xi) \big|,$$

for $\xi \in T_{\mathbf{X}} \mathcal{M}_{\mathbf{r}}$ and $h \geq 0$. Then $m_{\mathbf{X}}$ is an order-q model in $\mathbf{X}$ if and only if

$$e(\xi, h) = \mathcal{O}\big(h^{q+1}\big), \quad \text{for all } \xi \in T_{\mathbf{X}} \mathcal{M}_{\mathbf{r}}.$$

In Figures 4.3 and 4.4, we test the model orders of (4.8)–(4.10). We generate random tensors $\mathbf{B}_1, \ldots, \mathbf{B}_{1000} \in \mathbb{R}^{10 \times 10 \times 10}$ with normally distributed entries and project them onto a given tangent space of $\mathcal{M}_{(3,3,3)}$ to get $\xi_i = \mathrm{P}_{\mathbf{X}}(\mathbf{B}_i)$. We normalize the resulting vectors to get $\|\xi_i\| = 1$. We compute the errors $e(\xi_i, 2^{-j})$ for $j = 0, \cdots, 10$, and plot the geometric mean of the factors $(\xi_i, 2^{-(j+1)})/(\xi_i, 2^{-j})$ over all i. The first columns contain the results for a stationary point of f, i. e. $\|\operatorname{grad} f(\mathbf{X}^*)\| = 0$, the second columns contain the results for an arbitrary point on the manifold with $\|\operatorname{grad} f(\mathbf{X})\| \neq 0$. The first, second and third rows contain results for different sampling sizes, with $|\Omega| = 10, 100, 1000$, respectively. Note that $|\Omega| = 1000$ represents full sampling, i. e. vector approximation. We write $f(\mathbf{X}) = 0$ whenever the function value computed is smaller that the machine precision of 10^{-16}.

We observe that the model function $m_{\mathbf{X}}^{\mathrm{SD}}$, indeed, provides results of first order in all cases. The model function $m_{\mathbf{X}}^{\mathrm{N}}$ provides results of second order not only in critical points, as has been proved by theory, but also in general points on the manifold. This can be seen as an indication that the retraction by HOSVD preserves second-order information although we cannot prove this. We also observe that the Gauß–Newton type model function $m_{\mathbf{X}}^{\mathrm{GN}}$ gives second-order results whenever the curvature term is small enough, otherwise it is only a first-order model; this matches the theoretical predictions we made earlier. It is especially worth noting that, for $\mathbf{A} \in \mathcal{M}_{\mathbf{r}}$, a Gauß–Newton model is sufficient; however, this result is not robust if we add some noise. Note that in the cases where the blue curve cannot be seen in the plot, the models $m_{\mathbf{X}}^{\mathrm{GN}}$ and $m_{\mathbf{X}}^{\mathrm{N}}$ match almost exactly.

We also remark that in the case of exact tensor reconstruction, i. e. $\mathbf{A} \in \mathcal{M}_{\mathbf{r}}$ and $|\Omega| = \prod_i n_i$ (the lower-left plot in Figure 4.4), both $m_{\mathbf{X}}^{\mathrm{N}}$ and $m_{\mathbf{X}}^{\mathrm{GN}}$ seem to be models of order 3, which means that the third-order term in the Taylor expansion of f vanishes.

This may be attributed to a possible symmetry of f around the local minimizer $\mathbf{X}^* = \mathbf{A}$ in this case, i. e. $f(\mathrm{Exp}_{\mathbf{X}^*}(\xi)) = f(\mathrm{Exp}_{\mathbf{X}^*}(-\xi))$, where Exp denotes the exponential map. This means that the odd-exponent terms in the Taylor expansion are equal to zero. However, we cannot verify this theoretically as we do not have a closed-form expression for the exponential map on $\mathcal{M}_{\mathbf{r}}$.

Since a CG iteration just requires a fixed number of matrix-vector products, the total cost of the trust-region method with exact Hessian evaluation is given by

$$\mathcal{O}\big(K_{\max}(r^d(|\Omega| + n) + r^{d+1})\big)$$

In general, we cannot rule out Algorithm 3.20 converging to a nonregular minimum if $|\Omega| < \dim(\mathcal{M}_{\mathbf{r}})$. If this causes problems, we can enforce positive-definiteness of the Hessian by considering a cost function regularized with an identity term

$$f_\mu(\mathbf{X}) = \frac{1}{2}\big\| \mathrm{P}_\Omega \, \mathbf{X} - \mathrm{P}_\Omega \, \mathbf{A}\big\|^2 + \frac{\mu}{2}\|\mathbf{X}\|^2,$$

for some $\mu > 0$. However, such a problem may not be well-posed since there is not enough information provided to recover $\mathbf{X}$ in a meaningful way. Moreover, in our practical experiments we did not have a need to use this regularization.

4.4 Numerical Experiments

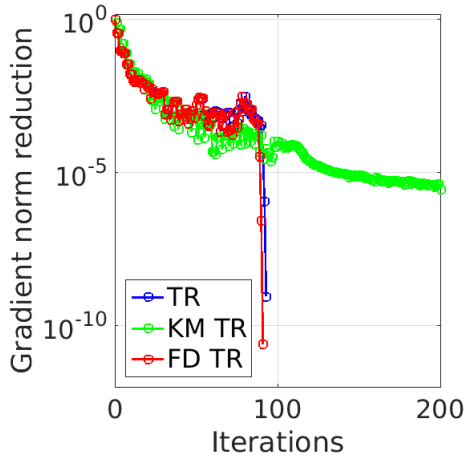
We implemented our method in MATLAB version 2015b using the Tensor Toolbox version 2.6 [5, 6] for the basic tensor arithmetic and Manopt version 3.0 [13] for handling the Riemannian trust-region scheme. All tests were performed on a quad-core Intel i7-2600 CPU with 8 GB of RAM running 64-Bit Ubuntu 16.04 Linux. Stated calculation times are wall-clock times, excluding the set-up time of the problem.

In Algorithm 3.20, we choose the standard parameters $\bar{\Delta} = \dim(\mathcal{M}_{\mathbf{r}})$, $\Delta_0 = \bar{\Delta}/8$, $\rho' = 0.1$. The initial guess $\mathbf{X}_0$ is generated randomly by a uniform distribution on $(0, 1)$ for each entry in the factors in the Tucker decomposition. We apply a QR factorization in each mode to ensure that the basis matrices are orthogonal. The sampling set Ω is chosen from a uniform distribution on the index set.

In Subsection 4.4.1 we test our algorithm for artificial, uniformly distributed data. Then we apply it to data from survey statistics in Subsection 4.4.2 and to hyperspectral imaging data in Subsection 4.4.3.

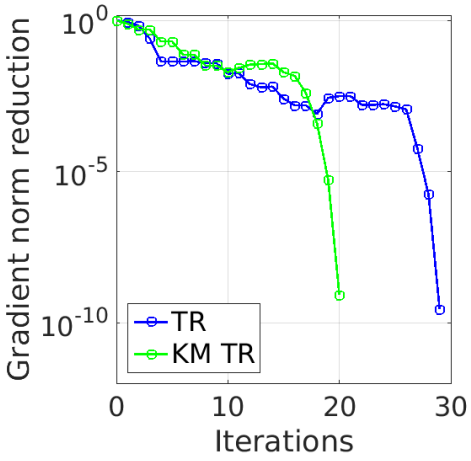
4.4.1 Uniformly Distributed Random Data

We test the convergence behaviour of Algorithm 3.20 for the recovery of a partially known tensor $\mathbf{A}$ with uniformly distributed entries; the results are shown in Figure 4.5. Two exemplary cases are shown to point out the properties of our method. We observe that our trust-region method (TR) with exact Hessian computation yields superlinear convergence after a small number of iterations in the cases observed here. The finite difference Hessian approximation (FD TR) as introduced in [11] shows similar behaviour;



	Time	$\|\mathbf{X}-\mathbf{A}\|/\|\mathbf{A}\|$	CG it.
TR	44.2	9.6×10^{-10}	8.6
KM TR	226.6	0.14	60.2
FD TR	72.8	3.5×10^{-10}	8.6

(a): $n_1 = 80$, $n_2 = n_3 = 20$, $r_1 = 10$, $r_2 = r_3 = 5$



	Time	$\|\mathbf{X}-\mathbf{A}\|/\|\mathbf{A}\|$	CG it.
TR	6.6	1.4×10^{-2}	4.2
KM TR	0.8	1.9×10^{-2}	3.3
FD TR	10.9	1.4×10^{-2}	4.2

(b): $n_1 = 20$, $n_2 = 30$, $n_3 = 40$, $r_i = 0.1 n_i \ \forall i$

Figure 4.5: Convergence of Riemannian trust region methods for (4.1). The case (b) includes noise.

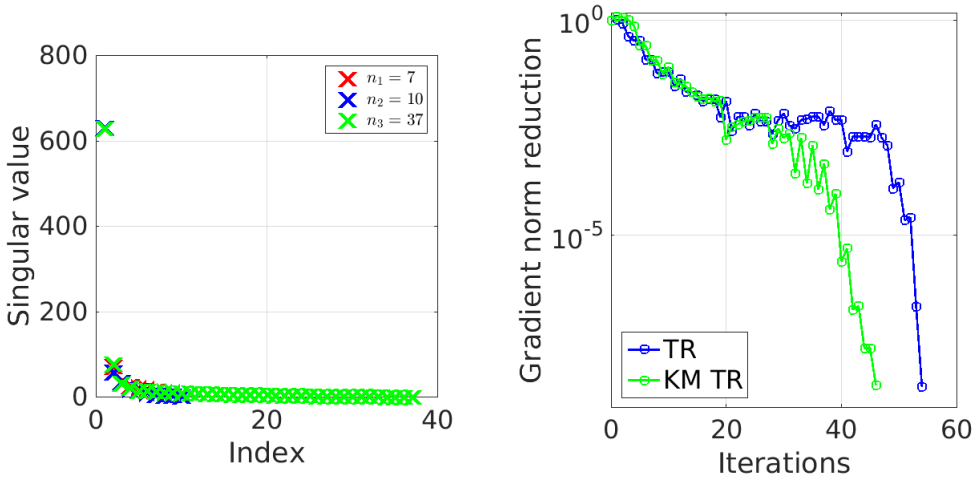


Figure 4.6: Left: Singular values of the data set [60]; right: convergence of Riemannian trust region methods for tensor completion with $|\Omega| = 0.5 \times \prod_i n_i$ and $\mathbf{r} = (5, 5, 5)$. The recovery quality is given by $\|\mathbf{X} - \mathbf{A}\|/\|\mathbf{A}\| = 0.11$ for TR and $\|\mathbf{X} - \mathbf{A}\|/\|\mathbf{A}\| = 0.32$ for KM TR; the runtime is 28.1 for TR and 4.1 for KM TR.

in case (b) it is not plotted since the iterates are not discernible from TR. In case (a), it is even slightly faster in terms of outer iterations; however, the exact method has superior runtime since a finite difference approximation is expensive. The state-of-the-art trust-region method [46] (KM TR) can slow down when the problem is ill-conditioned as is shown in case (a). Here, our method is more robust. In case (b), where a normally distributed noise tensor is added to the partially known tensor $\mathbf{A}$, our method, while slower, provides a better recovery of $\mathbf{A}$ as can be seen in the third column. The average number of iterations of the inner CG scheme (last column) is reasonable in both cases.

4.4.2 Survey Data

In survey statistics, data in the form of order-3 tensors arises in a natural way: for n_1 of individuals, n_2 properties are collected over n_3 time points; see, for example, [61]. We choose a standard data set [60], containing reading proficiency test measures of schoolchildren over a period of time. A typical problem in such data sets in practice is missing entries, resulting from nonresponse or failure to enter some of the data points correctly; see [67]. A typical application case is a sampling set greater or equal to half the total tensor size. As Figure 4.6 shows, data of this type shows rapidly decaying singular values, especially in the time mode ($i = 3$) and our trust-region method can be used to retrieve deleted data in a low-rank framework. Our trust-region method also converges superlinearly in this case. While our method is slower than the Kasai–Mishra trust-region scheme by a factor of 7 in this case, it provides a significantly better recovery

of the tensor $\mathbf{A}$.

4.4.3 Hyperspectral Image Data

As a final application, we consider the hyperspectral image 'Ribeira' from [25], which has been studied in the context of tensor completion by [57, 79]. This results in a tensors of size $1017 \times 1340 \times 33$, where each slice corresponds to an image of the same scene measured in a different wavelength. We assume the tensor to have low multilinear rank, see Figure 4.7. The reconstruction quality of tensor completion is illustrated in Figure 4.8. As can be seen in Table 4.1, the novel method outperforms the state of the art both in terms of runtime and reconstruction quality.

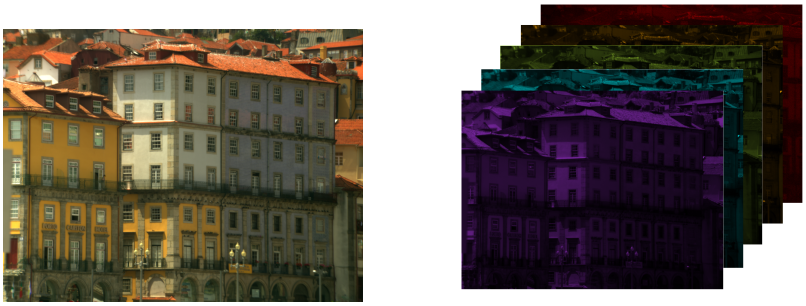


Figure 4.7: True-color representation (left) and symbolic representation of the hyperspectral 'Ribeira' image as a tensor (right).

Figure 4.8: The 18th slice of the hyperspectral image with 70 % of the information deleted (left) and reconstructed by RTR with $\mathbf{r} = (35, 35, 6)$.

	Time (s)	$\|\mathbf{X} - \mathbf{A}\|/\|\mathbf{A}\|$	TR it.
TR	1232	0.147	65
KM TR	2347	0.161	41

Table 4.1: TR iteration counts, CPU times and reconstruction quality for convergence to a relative gradient norm of 10^{-6} for the hyperspectral image 'Ribeira' for the novel trust-region method and the Kasai–Mishra method.

5 Optimal Control in Low-Rank Format

In this chapter, we will apply low-rank tensor numerical methods to optimal control of a fractional partial differential equation. In Subsection 5.1, we will give a brief introduction to optimal control. Section 5.2 describes the considered problem classes. Section 5.3 discusses the FEM/FDM discretization schemes, formulates the traditional Lagrange multiplies approach and describes the Kronecker product tensor structure in the discrete Laplacain type matrices in many dimensions. In Section 5.4 we discuss the application of rank truncation schemes to function-related matrices and tensors. Section 5.5 analyzes the tensor approximation of the inverse to the fractional Laplace operator and to some related matrix valued functions of fractional Laplacian in $\mathbb{R}^d$ arising in representation of the unknown control and design functions. Finally, in Section 5.6, we collect the results of numerical tests for 2D and 3D examples, which confirm the efficiency of the tensor approximation in the considered class of optimal control problems.

5.1 Introduction to Optimal Control and Literature Overview

Optimization problems with partial differential equations (PDEs) as constraints are well known in the mathematical literature. They admit a number of applications in various fields of natural sciences and have been studied for many years; see [84] for an introduction. The goal is to find a solution to a given PDE, which minimizes a given objective function; classical examples include *tracking-type* functionals, where we need to find a solution which matches a given profile as closely as possible.

Optimal control problems pose a major challenge from a computational point of view due to the complexity of constraint: evaluating the constraint requires the solution of a partial differential equation. Therefore, to make these problems tractable, specially tailored solvers are required.

In the classical sense, partial differential equations are given by local operators, i. e. only local information is required to evaluate the operator at a given point in the domain. When the PDE is discretized, the locality of the operator infers sparsity of the discrete operator—to evaluate the operator on a grid point, only the information at the neighboring grid points is needed. In this case, classical elliptic problem solvers for the forward problem can be modified to apply to the optimization problems, see [39] for an overview. Multigrid methods for elliptic equations are shown to be particularly efficient, since their computational complexity is linear in the number of degrees of freedom, i. e. in the number of grid points in the computational domain in $\mathbb{R}^d$, see [10, 9].

In recent literature, the study of problems with nonlocal constraints has attracted particular interest, where the operator is not differential but of integral type. The prototypical example of this kind of problems is the *fractional Laplacian* operator, which is

attained by taking a fractional power of the classical Laplacian operator, see, for example, [66, 63, 34, 27]. These problems pose an additional computational challenge: since local information is not sufficient for the evaluation of the operator, the discretized operator will be a dense matrix instead of a sparse one—if implemented in a straightforward way, even the simplest matrix-vector operations would have a quadratic complexity in the number of degrees of freedom in the computational box in $\mathbb{R}^d$. This would make these problems intractable even on moderately fine grids, especially in the three-dimensional case.

A number of approaches have been proposed in the literature to circumvent these difficulties, see papers [40, 34, 89, 22, 35] which consider approximation methods for fractional elliptic operator and [21, 90] related to time-dependent problems. An extension method has been proposed [15], which reduces a fractional Laplacian problem to a classical Laplacian problem in a higher-dimensional space and allows to make the problem tractable in some cases. Recently, a proof of concept for an optimal control solver based on the extension approach has been proposed, see [4].

However, we note that above approaches based on the conventional techniques of numerical analysis providing a linear scaling in the problem size at best, are exhibiting the exponential growth in the storage and computational complexity as $O(n^d)$ in the number of dimensions d, where n is the univariate grid size. A promising approach, which exploits the problem structure for the extraction of the numerical benefits from the specific features in the system of interest, is based on the concept of rank-structured tensor representation (approximation) of the target multivariate functions and operators, which allows to avoid this "curse of dimensionality".

In the last decades, extensive research efforts have been focused on different aspects of multilinear algebra, tensor calculus and related issues, see [18] for an overview. Recent tensor numerical methods appeared as a bridging of the well-known tensor decompositions from multilinear algebra with basic results in nonlinear approximation theory by using low-rank separable representations of the multidimensional functions and operators [27, 28]. In recent years, the development of low-rank tensor numerical methods has been a prior direction of mathematical research in scientific computing [50, 47, 59, 75]. The main idea of tensor methods is reducing the numerical solution of the multidimensional integral-differential equations to one-dimensional tensor product operations.

In this chapter, we introduce a tensor numerical method for the efficient numerical solution of the d-dimensional optimal control problems with fractional Laplacian type operators in constraints discretized on large spacial grids. We propose and analyze the approximate low-rank structure representations of functions of the fractional Laplacian $(-\Delta)^{\alpha}$, and its inverse $(-\Delta)^{-\alpha}$, $\alpha > 0$, in the bounded domain in $\mathbb{R}^d$, $d = 2, 3$, by using the canonical tensor format. There are many equivalent definitions of the fractional elliptic operator based on either integral or spectral formulation, see [66, 63, 34, 27]. Our tensor approach is based on the spectral decomposition of the target operators.

The functions of the finite element (finite difference) Laplacian operator on a tensor grid are diagonalized by using the fast Fourier transform (FFT) matrix and then a low-rank tensor approximation to the multidimensional core diagonal tensor is computed. In the three-dimensional case, the multigrid Tucker tensor decomposition [52] and Tucker-

to-canonical transform are employed [47]. A theoretical justification for the low-rank canonical and Tucker tensor approximation of functions of the discrete fractional Laplacian is sketched. We then show that these low-parametric representations transfer to the solution operator of an optimal control problem governed by the fractional Laplacian. In this way, the spacial dimensions can be approximately separated to admit a low-rank tensor structure in the solution vector. Our approach allows to make the solution of the considered optimal control problems computationally tractable in both the two- and three-dimensional cases. We justify that optimal control problems of this class can be solved with a cost which is only slightly larger than linear in the number of grid points in one spacial dimension, independently of the number of spacial dimensions, which is a considerable improvement over classical solvers, whose cost scales exponentially in the spacial dimension. We provide convincing numerical results to support our theoretical reasoning.

Notice that the low-rank tensor methods in the context of fractional time-dependent optimal control based on a one-dimensional fractional Laplacian operator have been recently reported in [21].

5.2 Problem Setting

Our goal is the construction of fast solution schemes for solving the control problems with d-dimensional fractional elliptic operators in the constraint. For this reason, we restrict ourselves to the case of rectangular domains and to the class of generalized Laplacian type elliptic operators with separable coefficients.

Given the design function $y_\Omega \in L^2(\Omega)$ on $\Omega := (0,1)^d$, $d = 1, 2, 3$, first, we consider the optimization problem for the cost functional

$$\min_{y,u} J(y,u) := \int_\Omega (y(x) - y_\Omega(x))^2 \,\mathrm{d}x + \frac{\gamma}{2} \int_\Omega u^2(x) \,\mathrm{d}x, \tag{5.1}$$

constrained by the elliptic boundary value problem in Ω for the state variable $y \in H_0^1(\Omega)$,

$$\mathcal{A}y := -\nabla^T \cdot \mathbb{A}(x) \nabla y = \beta u, \quad x \in \Omega,\ u \in L_2(\Omega), \quad \beta > 0, \tag{5.2}$$

endorsed with the homogeneous Dirichlet boundary conditions i. e. $y|_{\partial\Omega} = 0$. The coefficient matrix $\mathbb{A}(x) \in \mathbb{R}^{d\times d}$ is supposed to be symmetric, positive definite and uniformly bounded in Ω with positive constants $c > 0$ and $C > 0$, i. e.

$$c\, I_d \le \mathbb{A}(x) \le C\, I_d.$$

Under the above assumptions the associated bilinear form

$$A(u,v) = \int_\Omega \mathbb{A}(x) \nabla u(x) \cdot \nabla v(x) \,\mathrm{d}x$$

defined on $V \times V$, $V := \{v \in H_0^1(\Omega)\}$ is symmetric, coercive and bounded on V with the same constants c and C.

In what follows, we describe a tensor method for the fast numerical solution of the optimization problem with the generalized constraints

$$\mathcal{A}^{\alpha} y = \beta u(x), \quad x \in \Omega, \tag{5.3}$$

such that for $0 < \alpha \leq 1$, the fractional power of the elliptic operator $\mathcal{A}$ is defined by

$$\mathcal{A}^{\alpha} y(x) = \sum_{i=1}^{\infty} \lambda_i^{\alpha} c_i \psi_i(x), \quad y = \sum_{i=1}^{\infty} c_i \psi_i(x), \tag{5.4}$$

where $\{\psi_i(x)\}_{i=1}^{\infty}$ is the set of L_2-orthogonal eigenfunctions of the symmetric, positive definite operator $\mathcal{A}$, while $\{\lambda_i\}_{i=1}^{\infty}$ are the corresponding (real and positive) eigenvalues.

Notice that the elliptic operator inverse $\mathcal{A}^{-1} = \mathcal{T} : L_2(\Omega) \to V$, where $\mathcal{A} = \mathcal{T}^{-1}$, provides an explicit representation for the state variable, $y = \beta \mathcal{T} u = \beta \mathcal{A}^{-1} u$ in case (5.2), while in the general case (5.3) we have

$$y = \beta \mathcal{T}^{\alpha} u = \beta \mathcal{A}^{-\alpha} u. \tag{5.5}$$

Here $\mathcal{T}$ is a compact, symmetric and positive definite operator on $L_2(\Omega)$ and its eigenpairs $\{\psi_i, \mu_i\}$, $i = 1, \ldots, \infty$, provide an orthonormal basis for $L_2(\Omega)$. Clearly, we have $\lambda_i^{-1} = \mu_i$.

There are several equivalent representations (definitions) for the fractional powers of the symmetric, positive definite operators $\mathcal{A}^{\alpha}$ and $\mathcal{A}^{-\alpha}$ with $0 < \alpha \leq 1$, see for example the survey paper [63]. It should be noted that equivalence only holds when viewing the operators as defined on $\mathbb{R}^d$; when we consider them on bounded domains with some boundary conditions, it does not hold in general, cf. [66]. In particular, the Dunford–Taylor contour integral and the Laplace transform integral representations could be applied. In the presented computational schemes based on low rank tensor decompositions, we apply the Laplace transform integral representation. For $\alpha > 0$, the integral representation based on the Laplace transform

$$\mathcal{A}^{-\alpha} = \frac{1}{\Gamma(\alpha)} \int_0^{\infty} t^{\alpha - 1} e^{-t\mathcal{A}} \, dt \tag{5.6}$$

suggests the numerical schemes for low rank canonical tensor representation of the operator (matrix) $\mathcal{A}^{-\alpha}$ by using the sinc quadrature approximations for the integral on the real axis [28]. The efficiency of this approach is based on the exponentially fast convergence of the sinc quadratures on a class of analytic functions.

Furthermore, for $\alpha > 0$ the Dunford–Taylor (or Dunford–Cauchy) contour integral representation reads (see for example [40, 34, 27])

$$\mathcal{A}^{-\alpha} = \frac{1}{2\pi i} \int_{\mathcal{G}} z^{-\alpha} (\mathcal{A} - z\mathcal{I})^{-1} \, \mathrm{d}z, \tag{5.7}$$

where the contour $\mathcal{G}$ in the complex plane includes the spectrum of operator (matrix) $\mathcal{A}$. This representation applies to any $u \in L_2(\Omega)$, and it allows to define the negative

fractional powers of an elliptic operator as a finite sum of elliptic resolvents $\mathcal{R}_z(\mathcal{L}) = (z\mathcal{I} - \mathcal{L})^{-1}$ by application of certain quadrature approximations, see also [34, 27, 28]. This opens the way for multigrid-type or $\mathcal{H}$-matrix (see [27]) schemes approximating the fractional powers of elliptic operator with variable coefficients of rather general form.

It is worth noting that both integral representations (5.6) and (5.7) can be applied to a rather general class of analytic functions of operator $f(\mathcal{A})$, including the case $f(\mathcal{A}) = \mathcal{A}^{-\alpha}$, see [40, 27, 28].

The constraint equation (5.5) allows to derive the Lagrange equation for the control u in explicit form as follows (see Section 5.3 concerning the Lagrange equations)

$$\big(\beta\mathcal{A}^{-\alpha} + \tfrac{\gamma}{\beta}\mathcal{A}^{\alpha}\big)u = y_{\Omega}, \tag{5.8}$$

for some positive constants $\beta > 0$ and $\gamma > 0$. This equation implies the following representation for the state variable

$$y = \beta\mathcal{A}^{-\alpha}u = \big(\mathcal{I} + \tfrac{\gamma}{\beta^2}\mathcal{A}^{2\alpha}\big)^{-1} y_{\Omega}. \tag{5.9}$$

The practically interesting range of parameters includes the case $\beta = O(1)$ for small values of $\gamma > 0$. Our tensor numerical scheme is focused on the solution of equations (5.8) and (5.9) that include the nonlocal operators of "integral-differential" type. The efficiency of the rank-structured tensor approximations presupposes that the design function in the right-hand side of these equations, $y_{\Omega}(x_1, x_2, x_3)$, allows a low-rank separable approximation.

Since we aim for the low-rank (approximate) tensor representation of all functions and operators involved in the above control problem, we further assume that the diffusion coefficient matrix takes a diagonal form

$$\mathbb{A}(x) = \mathrm{diag}\{a_1(x_1), a_2(x_2)\}, \quad a_\ell(x_\ell) > 0, \quad \ell = 1, 2,$$

in the case $d = 2$, and similarly, for $d = 3$,

$$\mathbb{A}(x) = \mathrm{diag}\{a_1(x_1), a_2(x_2), a_3(x_3)\}, \quad a_\ell(x_\ell) > 0, \quad \ell = 1, 2, 3. \tag{5.10}$$

In what follows, we consider a discrete matrix formulation of the optimal control problem (5.1), (5.3) based on the FEM/FDM discretization of d-dimensional elliptic operators defined on the uniform $n_1 \times \cdots \times n_d$ tensor grid in Ω.

The fractional elliptic operator $\mathcal{A}^{\alpha}$ is approximated by its FEM/FDM representation $(\mathcal{A}_h)^{\alpha}$, subject to the homogeneous Dirichlet boundary condition, where the operator (matrix) $(\mathcal{A}_h)^{\alpha}$ is defined as in (5.4), where the eigenpairs for the corresponding grid discretization $\mathcal{A}_h$ of the elliptic operator $\mathcal{A}$ are used. Here $h = h_\ell = 1/n_\ell$ is the univariate mesh parameter. The FEM/FDM approximation theory for fractional powers of elliptic operator was presented in [22], see also the literature therein.

5.3 Optimality Conditions and Representations in a Low-Rank Format

The solution of problem (5.1) with constraint (5.3) requires solving for the necessary optimality conditions. In this section, we will derive these conditions based on a discretize-then-optimize-approach. Then, we will discuss how the involved discretized operators can be applied efficiently in a low-rank format, and how this can be used to design a preconditioned conjugate gradient (PCG) scheme for the necessary optimality conditions.

In Subsection 5.3.1, we will calculate the optimality conditions for the optimal control problem based on the formale Lagrange principle. In Subsection 5.3.2, we will explain the multiplication of multidimensional matrices and vectors given in a low-rank format. Finally, in Subsection 5.3.3, we will present a preconditioned conjugate gradient scheme tailored for data in the low-rank tensor format.

5.3.1 Discrete Optimality Conditions

We consider the discretized version of the control problem (5.1)–(5.3). We assume we have a uniform grid in each space dimension, i. e. we have $N = n_1 n_2$ (for $d = 2$) or $N = n_1 n_2 n_3$ (for $d = 3$) grid points. We will denote the discretized state y, design y_Ω and control u by vectors $\mathbf{y}, \mathbf{y}_\Omega, \mathbf{u} \in \mathbb{R}^N$, respectively. For simplicity, we assume that we use the same approximation for all quantities.

Then, the discrete problem is given as

$$\begin{aligned} \min_{\mathbf{y},\mathbf{u}} & \frac{1}{2}(\mathbf{y}-\mathbf{y}_\Omega)^{\mathrm{T}} M(\mathbf{y}-\mathbf{y}_\Omega) + \frac{\gamma}{2}\mathbf{u}^{\mathrm{T}} M\mathbf{u} \\ \text{s. t. } & A^\alpha \mathbf{y} = \beta M \mathbf{u}, \end{aligned}$$

where $A = \mathcal{A}_h$ denotes a discretization of the elliptic operator $\mathcal{A}$ by finite elements or finite differences. The matrix M will be a mass matrix in the finite element case and simply the identity matrix in the finite difference case.

For the discrete adjoint $\mathbf{p}$, define the Lagrangian function

$$L(\mathbf{y},\mathbf{u},\mathbf{p}) = \frac{1}{2}(\mathbf{y}-\mathbf{y}_\Omega)^{\mathrm{T}} M(\mathbf{y}-\mathbf{y}_\Omega) + \frac{\gamma}{2}\mathbf{u}^{\mathrm{T}} M\mathbf{u} + \mathbf{p}^{\mathrm{T}}(A^\alpha \mathbf{y} - \beta M\mathbf{u}), \tag{5.11}$$

and compute the necessary first order conditions, given by the Karush-Kuhn-Tucker (KKT) system,

$$\begin{bmatrix} M & O & A^\alpha \\ O & \gamma M & -\beta M \\ A^\alpha & -\beta M & O \end{bmatrix} \begin{pmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{y}_\Omega \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}. \tag{5.12}$$

We can solve the state equation for y, getting

$$\mathbf{y} = \beta A^{-\alpha}\mathbf{u},$$

and the design equation for p getting

$$\mathbf{p} = \tfrac{\gamma}{\beta}\mathbf{u}.$$

Hence the adjoint equation gives us an equation for the control $\mathbf{u}$, namely

$$\big(\beta A^{-\alpha} + \tfrac{\gamma}{\beta}A^{\alpha}\big)\mathbf{u} = \mathbf{y}_{\Omega}. \tag{5.13}$$

5.3.2 Matrix-Vector Multiplication in the Low-Rank Format

Now we derive a decomposition of the discrete Laplacian A which is compatible with low-rank data. Let I_ℓ denote the identity matrix, and A_ℓ the discretized one-dimensional Laplacian on the given grid in the ℓth mode, then we have

$$A = A_1 \otimes I_2 \otimes I_3 + I_1 \otimes A_2 \otimes I_3 + I_1 \otimes I_2 \otimes A_3. \tag{5.14}$$

If a matrix A has a decomposition as in (5.14), we will say that A has *Kronecker rank* equal to 3, i. e. to the number of Kronecker rank-one summands.

To obtain the matrices A_ℓ, we simply discretize the one-dimensional subproblems

$$\begin{aligned} -y''(x_\ell) &= \beta u(x_\ell), \\ y(0) &= 0 = y(1). \end{aligned}$$

Using a uniform grid with grid size h_ℓ, we obtain the discretizations

$$\underbrace{-\frac{1}{h_\ell}\begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}}_{=A_\ell} \begin{pmatrix} y_1^{(\ell)} \\ \vdots \\ y_{n_\ell}^{(\ell)} \end{pmatrix} = \beta \begin{pmatrix} u_1^{(\ell)} \\ \vdots \\ u_{n_\ell}^{(\ell)} \end{pmatrix}$$

The one-dimensional operator A_ℓ has an eigenvalue decomposition in the Fourier basis, i. e.

$$A_\ell = F_\ell^* \Lambda_\ell F_\ell.$$

In the case of homogeneous Dirichlet boundary conditions, the matrix F_ℓ defines the sin-Fourier (see, for example, [77, Section 12.3]) transform while $\Lambda_\ell = \operatorname{diag}\{\lambda_1^{(\ell)}, \ldots, \lambda_n^{(\ell)}\}$, where λ_k denote the eigenvalues of the univariate discrete Laplacian with Dirichlet boundary conditions. These are given by

$$\lambda_k = -\frac{4}{h_\ell^2}\sin^2\left(\frac{\pi k}{2(n_\ell+1)}\right) = -\frac{4}{h_\ell^2}\sin^2\left(\frac{\pi k h_\ell}{2}\right). \tag{5.15}$$

Thus, using the properties of the Kronecker product, we can write the first summand in (5.14) as

$$\begin{aligned} A_1 \otimes I_2 \otimes I_3 &= (F_1^*\Lambda_1 F_1) \otimes (F_2^* I_2 F_2) \otimes (F_3^* I_3 F_3) \\ &= (F_1^* \otimes F_2^* \otimes F_3^*)(\Lambda_1 \otimes I_2 \otimes I_3)(F_1 \otimes F_2 \otimes F_3). \end{aligned}$$

The decomposition of the second and third summand works analogously, thus we can write equation (5.14) as

$$\begin{aligned} A =&(F_1^* \otimes F_2^* \otimes F_3^*)(\Lambda_1 \otimes I_2 \otimes I_3)(F_1 \otimes F_2 \otimes F_3) \\ &+ (F_1^* \otimes F_2^* \otimes F_3^*)(I_1 \otimes \Lambda_2 \otimes I_3)(F_1 \otimes F_2 \otimes F_3) \\ &+ (F_1^* \otimes F_2^* \otimes F_3^*)(I_1 \otimes I_2 \otimes \Lambda_3)(F_1 \otimes F_2 \otimes F_3) \\ =& (F_1^* \otimes F_2^* \otimes F_3^*)\big((\Lambda_1 \otimes I_2 \otimes I_3) + (I_1 \otimes \Lambda_2 \otimes I_3) + (I_1 \otimes I_2 \otimes \Lambda_3)\big)(F_1 \otimes F_2 \otimes F_3). \end{aligned} \tag{5.16}$$

The above expression gives us the eigenvalue decomposition, which can be used to efficiently compute functions of A.

In the case $d = 2$, the expression simplifies to

$$\begin{aligned} A &= (F_1^* \otimes F_2^*)(\Lambda_1 \otimes I_2)(F_1 \otimes F_2) + (F_1^* \otimes F_2^*)(I_1 \otimes \Lambda_2)(F_1 \otimes F_2) \\ &= (F_1^* \otimes F_2^*) \underbrace{\big((\Lambda_1 \otimes I_2) + (I_1 \otimes \Lambda_2)\big)}_{=:\Lambda}(F_1 \otimes F_2). \end{aligned} \tag{5.17}$$

Note that expression (5.17) gives us the eigenvalue decomposition of A. Therefore, for a function $\mathcal{F}$ applied to A, we get

$$\mathcal{F}(A) = (F_1^* \otimes F_2^*)\mathcal{F}(\Lambda)(F_1 \otimes F_2). \tag{5.18}$$

Now assume that $\mathcal{F}(A)$ may be expressed approximately by a linear combination of Kronecker rank-one operators. Then, to approximate $\mathcal{F}(A)$ it is sufficient to approximate the diagonal matrix $\mathcal{F}(\Lambda)$. Assume we have a decomposition

$$\mathcal{F}(\Lambda) = \sum_{k=1}^{R} \operatorname{diag}\big(\mathbf{u}_1^{(k)} \otimes \mathbf{u}_2^{(k)}\big),$$

with vectors $\mathbf{u}_i^{(k)} \in \mathbb{R}^{n_i}$ and $R \ll \min(n_1, n_2)$. Now let $\mathbf{x} \in \mathbb{R}^N$ be a vector given in a low rank format, i. e.

$$\mathbf{x} = \sum_{j=1}^{S} \mathbf{x}_1^{(j)} \otimes \mathbf{x}_2^{(j)},$$

with vectors $\mathbf{x}_i^{(j)} \in \mathbb{R}^{n_i}$ and $S \ll \min(n_1, n_2)$. Then, using (2.15) we can compute a matrix-vector product

$$\begin{aligned} \mathcal{F}(A)\mathbf{x} &= (F_1^* \otimes F_2^*)\Big(\sum_{k=1}^{R} \operatorname{diag}\big(\mathbf{u}_1^{(k)} \otimes \mathbf{u}_2^{(k)}\big)\Big)(F_1 \otimes F_2)\Big(\sum_{j=1}^{S} \mathbf{x}_1^{(j)} \otimes \mathbf{x}_2^{(j)}\Big) \\ &= \sum_{k=1}^{R}\sum_{j=1}^{S} F_1^*\big(\mathbf{u}_1^{(k)} \odot F_1\mathbf{x}_1^{(j)}\big) \otimes F_2^*\big(\mathbf{u}_2^{(k)} \odot F_2\mathbf{x}_2^{(j)}\big), \end{aligned} \tag{5.19}$$

where $\odot$ denotes the componentwise (Hadamard) product.

Using the sin-FFT, expression (5.19) can be computed in factored form in $\mathcal{O}(RSn \log n)$ flops, where $n = \max(n_1, n_2)$.

In the case $d = 3$, with completely analogous reasoning, equation (5.19) becomes

$$\mathcal{F}(A)\mathbf{x} = \sum_{k=1}^{R} \sum_{j=1}^{S} F_1^* \big(\mathbf{u}_1^{(k)} \odot F_1 \mathbf{x}_1^{(j)}\big) \otimes F_2^* \big(\mathbf{u}_2^{(k)} \odot F_2 \mathbf{x}_2^{(j)}\big) \otimes F_3^* \big(\mathbf{u}_3^{(k)} \odot F_3 \mathbf{x}_3^{(j)}\big), \quad (5.20)$$

and similar in the case of $d > 3$.

5.3.3 The Low-Rank PCG Scheme

Algorithm 5.1 Preconditioned CG method in low-rank format.

Input: Rank truncation procedure `trunc`, rank tolerance parameter ε, linear function in low-rank format `fun`, preconditioner in low rank format `precond`, right-hand side tensor $\mathbf{B}$, initial guess $\mathbf{X}^{(0)}$, maximal iteration number $k_{\max}$.

$\mathbf{R}^{(0)} \leftarrow \mathbf{B} - \texttt{fun}(\mathbf{X}^{(0)})$
$\mathbf{Z}^{(0)} \leftarrow \texttt{precond}(\mathbf{R}^{(0)})$
$\mathbf{Z}^{(0)} \leftarrow \texttt{trunc}(\mathbf{Z}^{(0)}, \varepsilon)$
$\mathbf{P}^{(0)} \leftarrow (\mathbf{Z}^{(0)})$
$k \leftarrow 0$
repeat
  $\mathbf{S}^{(k)} \leftarrow \texttt{fun}(\mathbf{P}^{(k)})$
  $\mathbf{S}^{(k)} \leftarrow \texttt{trunc}(\mathbf{S}^{(k)}, \varepsilon)$
  $\alpha_k \leftarrow \frac{\langle \mathbf{R}^{(k)}, \mathbf{Z}^{(k)} \rangle}{\langle \mathbf{P}^{(k)}, \mathbf{S}^{(k)} \rangle}$
  $\mathbf{X}^{(k+1)} \leftarrow \mathbf{X}^{(k)} + \alpha_k \mathbf{P}^{(k)}$
  $\mathbf{X}^{(k+1)} \leftarrow \texttt{trunc}(\mathbf{X}^{(k+1)}, \varepsilon)$
  $\mathbf{R}^{(k+1)} \leftarrow \mathbf{R}^{(k)} - \alpha_k \mathbf{S}^{(k)}$
  $\mathbf{R}^{(k+1)} \leftarrow \texttt{trunc}(\mathbf{R}^{(k+1)}, \varepsilon)$
  if $\mathbf{R}^{(k+1)}$ is sufficiently small **then**
    return $\mathbf{X}^{(k+1)}$
    break
  end if
  $\mathbf{Z}^{(k+1)} \leftarrow \texttt{precond}(\mathbf{R}^{(k+1)})$
  $\mathbf{Z}^{(k+1)} \leftarrow \texttt{trunc}(\mathbf{Z}^{(k+1)}, \varepsilon)$
  $\beta_k \leftarrow \frac{\langle \mathbf{R}^{(k+1)}, \mathbf{Z}^{(k+1)} \rangle}{\langle \mathbf{Z}^{(k)}, \mathbf{R}^{(k)} \rangle}$
  $\mathbf{P}^{(k+1)} \leftarrow \mathbf{Z}^{(k+1)} + \beta_k \mathbf{P}^{(k)}$
  $\mathbf{P}^{(k+1)} \leftarrow \texttt{trunc}(\mathbf{P}^{(k+1)}, \varepsilon)$
  $k \leftarrow k + 1$
until $k = k_{\max}$

Output: Solution $\mathbf{X}$ of $\texttt{fun}(\mathbf{X}) = \mathbf{B}$

For operators `func` and `precond` given in a low-rank format, such as (5.19) (for $d = 2$) or (5.20) (for $d = 3$), Krylov subspace methods can be applied very efficiently, since

they only require matrix-vector products. The formulation of the PCG method in Algorithm 5.1 is independent of d, as long as an appropriate rank truncation procedure `trunc` is chosen.

As the rank truncation procedure, in our implementation we apply the reduced SVD algorithm in 2D case and the RHOSVD based canonical-to-Tucker-to-canonical algorithm in the 3D case (see [52]) as described in the next section. One should take care that the convergence tolerance in lines 14–16 should be small enough in terms of the rank truncation tolerance (i. e. $\varepsilon_{\text{conv}} \geq c\,\varepsilon_{\text{trunc}}$, for some constant c), such that CG convergence is not hindered by the rank truncation.

5.4 Tensor Decomposition of Function Related Tensors

An integral part of the preconditioned CG scheme in Algorithm 5.1 is the function `trunc`, which truncates the rank of a given tensor in the CP format. In Subsection 5.4.1 we will discuss the case $d = 2$, and in Subsection 5.4.2 the case $d = 3$.

5.4.1 Reduced SVD of a Rank-r Matrix

Assume we have a rank-s matrix $A \in \mathbb{R}^{m\times n}$, given in factored format $A = UV^{\mathrm{T}}$, with $U \in \mathbb{R}^{m\times s}$ and $V \in \mathbb{R}^{n\times s}$. We are interested in the best rank-r approximation with $r < s$. Note that this is non-trivial if the columns of U and V do not coincide with the singular vectors of A. However, it can be done efficiently in factored form, without computing the full A, using the *reduced SVD* algorithm.

First, we perform the QR decomposition for the factor matrices to get

$$U = Q_U R_U,\ V = Q_V R_V,$$

with matrices $Q_U \in \mathbb{R}^{m\times s}$ and $Q_V \in \mathbb{R}^{n\times s}$ with othonormal columns, and upper-triangular matrices $R_U \in \mathbb{R}^{s\times s}$ and $R_V \in \mathbb{R}^{s\times s}$..

Now, we compute the SVD of the core matrix $R_U R_V^{\mathrm{T}} \in \mathbb{R}^{s\times s}$ to get

$$R_U R_V^{\mathrm{T}} = U_R \Sigma_R V_R^{\mathrm{T}},$$

with the diagonal matrix $\Sigma_R = \operatorname{diag}(\sigma_1, \ldots, \sigma_s)$, $\sigma_1 \geq \cdots \geq \sigma_s > 0$, and orthogonal matrices $U_R, V_R \in \mathbb{R}^{s\times s}$. Then, the best rank-$r$ approximation of $R_U R_V^{\mathrm{T}}$ is given by the truncated SVD; namely, we take the first r diagonal entries of Σ_R to get $\Sigma_r = \operatorname{diag}(\sigma_1, \ldots, \sigma_s)$, and the first r columns of U_R and V_R to get U_r and V_r, respectively.

Then the best rank-s approximation to A is given by

$$A_r = Q_U U_r \Sigma_r V_r^{\mathrm{T}} Q_V^{\mathrm{T}} = (Q_U U_r \Sigma^{1/2})(Q_V V_r \Sigma^{1/2})^{\mathrm{T}},$$

where $\Sigma^{1/2} = \operatorname{diag}(\sqrt{\sigma_1}, \ldots, \sqrt{\sigma_s})$ simply contains the square roots of the singular values. The approximation error is bounded by $\sqrt{\sum_{i=r+1}^{s} \sigma_i^2}$. The complexity of the above procedure scales linearly in the size of A, namely $\mathcal{O}(\max\{m, n\}r^2 + r^3)$.

5.4.2 Reduced Higher Order SVD of a Rank-r Tensor

For simplicity of presentation, we will only consider the case $d = 3$; the general case can be treated analogously. Assume we have a rank-s tensor $\mathbf{A}$ given in CP format

$$\mathbf{A} = \sum_{k=1}^{s} \lambda_k \, \mathbf{u}_k^{(1)} \circ \mathbf{u}_k^{(2)} \circ \mathbf{u}_k^{(3)} = [\![U_1, U_2, U_3]\!].$$

As we have seen in (2.13), it can be easily transformed into a Tucker tensor with rank $(r, r, r) \in \mathbb{R}^3$. However, the tensor rank r can be large and hence, the Tucker decomposition with a core of order $\mathcal{O}(r^3)$ will be large. When we know that we have (numerically) low-rank information, it makes sense to apply a scheme similar to the one in the previous subsection to reduce the rank. This is the idea of the *reduced higher-order SVD* (*RHOSVD*) introduced in [52].

For each of the factor matrices U_ℓ, we compute the SVD

$$U_\ell = V_\ell \Sigma_\ell W_\ell^{\mathrm{T}}, \ \ell = 1, 2, 3.$$

Given a rank threshold $\mathbf{r} = (r_1, r_2, r_3)$, we discard singular values and left singular vectors of U_ℓ to get $V_\ell^{(0)} \in \mathbb{R}^{n_\ell \times r_\ell}$, $\Sigma_\ell^{(0)} \in \mathbb{R}^{r_\ell \times r_\ell}$, and $W_\ell^{(0)} \in \mathbb{R}^{s \times r_\ell}$. Then, the RHOSVD of $\mathcal{A}$ is given by

$$\mathbf{A}_{\mathbf{r}}^{(0)} = \mathbf{C} \bigtimes_{\ell=1}^{3} \big(V_\ell^{(0)} \Sigma_\ell^{(0)} (W_\ell^{(0)})^{\mathrm{T}}\big),$$

where the core tensor $\mathbf{C}$ contains the weights λ_i on the hyperdiagonal:

$$c_{i_1 i_2 i_3} = \begin{cases} \lambda_i, \ \text{if } i = i_1 = i_2 = i_3, \\ 0, \ \text{else.} \end{cases}$$

Similar to the regular HOSVD in Subsection 2.2.2, the RHOSVD can be improved by an ALS iteration. We will briefly sketch the algorithm for the case $d = 3$. For more information, see, for example, [47, Section 3.3].

Starting with the RHOSVD of $\mathbf{A}$, we project the factor matrices U_ℓ onto the column basis of $V_\ell^{(0)}$:

$$\widetilde{U}_\ell \coloneqq (W_\ell^{(0)})^{\mathrm{T}} U_\ell = \Sigma_\ell^{(0)} (W_\ell^{(0)})^{\mathrm{T}}, \ \ell = 1, 2, 3.$$

Now we start the ALS iteration by computing the tensor $\mathbf{B}$ for $\ell = 1$:

$$\mathbf{B} = \sum_{k=1}^{s} \lambda_k \, \mathbf{u}_k^{(1)} \circ \tilde{\mathbf{u}}_k^{(2)} \circ \tilde{\mathbf{u}}_k^{(3)} \in \mathbb{R}^{n_1 \times r_2 \times r_3}.$$

We matricize B to get $B_{(1)} \in \mathbb{R}^{n_1 \times (r_2 r_3)}$, and compute the SVD

$$B_{(1)} = V_1^{(1)} \Sigma_\ell^{(1)} (W_\ell^{(1)})^{\mathrm{T}},$$

and truncate the left singular vectors to get the new basis matrix approximation $\widetilde{V}^{(1)} \in \mathbb{R}^{n_1 \times r_1}$. An analogous operation is performed for the cases of $\ell = 2$ and $\ell = 3$.

The ALS iteration is repeated at most $p_{\max}$ times, and we obtain the final basis matrix approximations $\widetilde{V}^{(1)}$, $\widetilde{V}^{(2)}$, and $\widetilde{V}^{(3)}$. Then the core tensor $\widetilde{\mathbf{C}}$ is obtained by applying the basis matrix $\widetilde{V}^{(3)}$ to the last iterate $\widetilde{\mathbf{B}}_3$, namely

$$\widetilde{\mathbf{C}} = \widetilde{\mathbf{B}}_3 \times_3 \widetilde{V}^{(3)}.$$

Now, we have applied the RHOSVD rank truncation to a tensor in the CP format and have obtained a tensor in the Tucker format. To get a tensor in the CP format again, we apply the *Tucker-to-canonical* transform introduced in [48, 51]. We state the main result.

Lemma 5.2 (Tucker–canonical approximation, [47, Lemma 3.6])**.** *Let*

$$\mathbf{A} = \mathbf{C} \bigtimes_{\ell=1}^{d} U_i$$

be a tensor in Tucker format. Assume that a best rank-r approximation $\mathbf{A}_r$ *to* $\mathbf{A}$ *exists. Then there exists a best rank-r approximation* $\mathbf{C}_r$ *to* $\mathbf{C}$*, such that*

$$\mathbf{A}_r = \mathbf{C}_r \bigtimes_{\ell=1}^{d} U_i.$$

Therefore, when we wish to transform a Tucker tensor to the CP format, it is sufficient to find a CP approximation to the core tensor (e. g. via the ALS routine introduced in Subsection 2.2.1). When we have found a decomposition

$$\mathbf{C} = [\![V_1, \ldots, V_d]\!], \tag{5.21}$$

the CP decomposition of the tensor $\mathbf{A}$ is given by

$$\mathbf{C} = [\![U_1, \ldots, U_d]\!], \tag{5.22}$$

with factor matrices $U_\ell = [V_\ell \mathbf{u}_1^{(\ell)} \cdots V_\ell \mathbf{u}_r^{(\ell)}]$.

5.5 Low-Rank Tensor Approximation of Functions of the Fractional Laplacian

In this section we analyze the rank structured tensor decompositions of various matrix-valued (or tensor-valued) functions on the discrete Laplacian arising in the solution of problems (5.8) and (5.9), including different combinations of fractional Laplacian in $\mathbb{R}^d$.

The elements of the core diagonal matrix Λ in (5.16) can be represented as a mode-three tensor $\mathbf{G} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, where

$$g_{i_1 i_2 i_3} = \lambda_{i_1} + \lambda_{i_2} + \lambda_{i_3},$$

implying that $\mathbf{G}$ has the exact rank-3 decomposition. In the case $d = 2$ we have similar two-term representation, $g_{i_1,i_2} = \lambda_{i_1} + \lambda_{i_2}$.

We consider the matrices A_1, A_2 and A_3 defined as the matrix-valued functions of the discrete Laplacain $\mathcal{A}_h$ by the equations

$$A_1 = \mathcal{A}_h^{-\alpha}, \tag{5.23}$$

$$A_2 = \mathcal{A}_h^{-\alpha} + \mathcal{A}_h^{\alpha}, \tag{5.24}$$

and

$$A_3 = \left(\mathcal{A}_h^{-\alpha} + \mathcal{A}_h^{\alpha}\right)^{-1} = A_2^{-1}, \tag{5.25}$$

respectively. It is worth noting that the matrix A_3 defines the solution operator in equation (5.8), which allows to calculate the optimal control in terms of the design function $\mathbf{y}_\Omega$ on the right-hand side of (5.8) as follows

$$\mathbf{u}^* = A_3 \mathbf{y}_\Omega. \tag{5.26}$$

Finally, the state variable is calculated by

$$\mathbf{y} = A_4 \mathbf{y}_\Omega, \quad \text{and} \quad A_4 = (I + \mathcal{A}_h^{2\alpha})^{-1} = \mathcal{A}_h^{\alpha} A_2^{-1}. \tag{5.27}$$

In the following numerical tests, we consider the behaviour of the SVD or Tucker decompositions for the corresponding multi-indexed core tensors/matrices G_1, G_2, G_3 and G_4 representing the matrix-valued functions A_1, A_2, A_3 and A_4 in the Fourier basis. It is well suited for the rank-structured algebraic operations since the d-dimensional Fourier transform matrix has Kronecker rank equals to one (see Section 5.3):

$$F = F_1 \otimes F_2 \otimes F_3.$$

Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues for the one-dimensional finite difference Laplacian in $H_0^1(0,1)$ discretized on the uniform grid with the mesh size $h = 1/(n+1)$.

In the 2D case, we analyze the singular value decomposition of the $n \times n$ core matrices

$$G_p = [g_{ij}^{(p)}], \quad p = 1,\, 2,\, 3,\, 4, \tag{5.28}$$

with entries defined by

$$g_{ij}^{(1)} = \frac{1}{(\lambda_i + \lambda_j)^{\alpha}}, \tag{5.29}$$

$$g_{ij}^{(2)} = \frac{1}{(\lambda_i + \lambda_j)^{\alpha}} + (\lambda_i + \lambda_j)^{\alpha}, \tag{5.30}$$

$$g_{ij}^{(3)} = \left(\frac{1}{(\lambda_i + \lambda_j)^{\alpha}} + (\lambda_i + \lambda_j)^{\alpha}\right)^{-1}, \tag{5.31}$$

$$g_{ij}^{(4)} = \left(1 + (\lambda_i + \lambda_j)^{2\alpha}\right)^{-1}, \tag{5.32}$$

In the 3D case, we consider the Tucker decomposition of the $n \times n \times n$ core tensor

$$\mathbf{G}_p = [g_{ijk}^{(p)}], \quad p = 1, 2, 3, 4 \tag{5.33}$$

with entries defined by

$$g^{(1)}_{ijk} = \frac{1}{(\lambda_i + \lambda_j + \lambda_k)^{\alpha}}, \tag{5.34}$$

$$g^{(2)}_{ijk} = \frac{1}{(\lambda_i + \lambda_j + \lambda_k)^{\alpha}} + (\lambda_i + \lambda_j + \lambda_k)^{\alpha}, \tag{5.35}$$

$$g^{(3)}_{ijk} = \left(\frac{1}{(\lambda_i + \lambda_j + \lambda_k)^{\alpha}} + (\lambda_i + \lambda_j + \lambda_k)^{\alpha} \right)^{-1}, \tag{5.36}$$

$$g^{(4)}_{ijk} = \left(1 + (\lambda_i + \lambda_j + \lambda_k)^{2\alpha}\right)^{-1}. \tag{5.37}$$

The error estimate for the rank decomposition of the matrices G_p and the respective mode-three tensors $\mathbf{G}_p$ can be derived based on the sinc-approximation theory. We consider the class of matrix valued functions of the discrete Laplacian, $A_1(\mathcal{A}_h), \ldots, A_4(\mathcal{A}_h)$, given by (5.23)–(5.25) and (5.27), respectively. In view of the FFT diagonalization, the tensor approximation problem is reduced to the analysis of the corresponding function related tensors $\mathbf{G}_1, \ldots, \mathbf{G}_4$ specified by multivariate functions of the discrete argument, $g_1, \ldots, g_4$, given by (5.34)–(5.37).

Our approach applies to the class of simplified "directionally fractional" Laplacian-type operators $\mathcal{A}_\alpha$ (with the diagonal coefficient as in (5.10)) obtained by the modification of (5.2) as follows

$$\mathcal{A}_\alpha := -\sum_{\ell=1}^{d} \left(\nabla_\ell^T \cdot a_\ell(x_\ell) \nabla_\ell\right)^{\alpha}, \quad \alpha > 0. \tag{5.38}$$

In this case the core tensors in (5.34)–(5.37) representing the operator $\mathcal{A}_\alpha$ in the Fourier basis are simplified to (for $a_\ell(x_\ell) = \text{const}$, $d = 3$)

$$\widetilde{g}^{(1)}_{ijk} = \frac{1}{\lambda_i^{\alpha} + \lambda_j^{\alpha} + \lambda_k^{\alpha}}, \tag{5.39}$$

$$\widetilde{g}^{(2)}_{ijk} = \frac{1}{\lambda_i^{\alpha} + \lambda_j^{\alpha} + \lambda_k^{\alpha}} + \lambda_i^{\alpha} + \lambda_j^{\alpha} + \lambda_k^{\alpha}, \tag{5.40}$$

$$\widetilde{g}^{(3)}_{ijk} = \left(\frac{1}{\lambda_i^{\alpha} + \lambda_j^{\alpha} + \lambda_k^{\alpha}} + \lambda_i^{\alpha} + \lambda_j^{\alpha} + \lambda_k^{\alpha} \right)^{-1}, \tag{5.41}$$

$$\widetilde{g}^{(4)}_{ijk} = \left(1 + (\lambda_i^{\alpha} + \lambda_j^{\alpha} + \lambda_k^{\alpha})^2\right)^{-1}, \tag{5.42}$$

correspondingly. Hence the rank behavior in the tensor decomposition of these discrete functions is completely similar to the case of Laplacian like operator in (5.34)–(5.37) corresponding to the fractional power $\alpha = 1$. This special case was considered in [21].

In the general case of variable coefficients $a_\ell(x_\ell) > 0$ in (5.38), the orthogonal matrices F_ℓ, $\ell = 1, \ldots, d$, of the univariate Fourier transforms in (5.16) should be substituted by the orthogonal matrices of the eigenvalue decomposition for the discretized elliptic operators $A_{h,\ell}$ (stiffness matrices) corresponding to $A_\ell = -\nabla_\ell^T \cdot a_\ell(x_\ell) \nabla_\ell$. The eigenvalues in

(5.39)–(5.42) are obtained from the solution of the eigenvalue problem $A_{h,\ell}\mathbf{u}_i = \lambda_{i,\ell}\mathbf{u}_i$, $\ell = 1, \ldots, d$.

The proof of the existence of the low rank canonical/Tucker decomposition of the core tensors $\mathbf{G}_p$, $p = 1, \ldots, 4$, is beyond the scope of this thesis. We refer to [37].

5.6 Numerics on Rank-Structured Approximations

In this section, we analyze the rank decomposition of all matrix entities involved in the solution operator (5.8). For the ease of exposition, in what follows, we set the model constants as $\beta = \gamma = 1$ and assume that $n_1 = n_2 = n_3$. Recall that $A = F^*GF$ with the notation $A = \mathcal{A}_h$, where $\mathcal{A}_h$ is the FDM approximation to the elliptic operator $\mathcal{A}$ and G is the diagonal core matrix represented in terms of eigenvalues of the discrete Laplacian $A = \mathcal{A}_h$. All numerical simulations are performed in MATLAB on a laptop. We will present 2D computations in Subsection 5.6.1 and 3D computations in Subsection 5.6.2.

First, we illustrate the smoothing properties of the elliptic operator $\mathcal{A}_h^{-\alpha}$ in 2D (or by the other words, the localization properties of the fractional operator $\mathcal{A}_h^{\alpha}$) in the equation for control depending on the fractional power $\alpha > 0$. Figures 5.1, 5.1, 5.3 and 5.4 represent the shape of the design function y_Ω and the corresponding optimal control $\mathbf{u}^*$ in the equation (5.26) computed for different values of the parameter α and for fixed grid size $n = 255$.

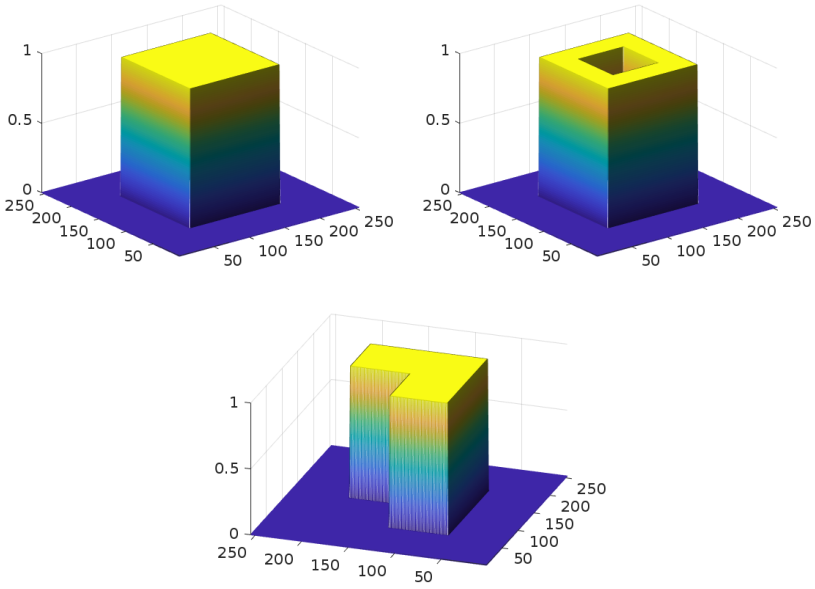


Figure 5.1: Shapes of the right-hand side y_Ω used in the 2D equation (5.26) computed with n=255.

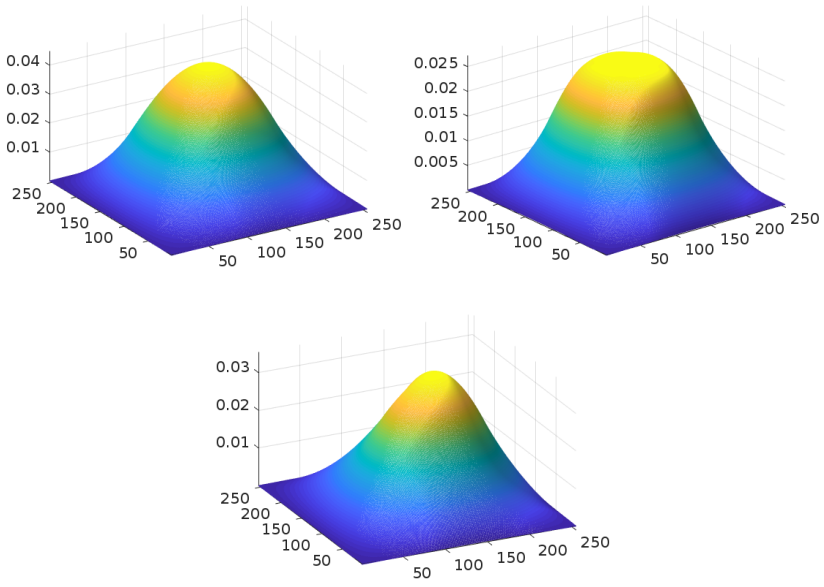


Figure 5.2: Solutions $\mathbf{u}^*$ for above right-hand sides y_Ω with $\alpha = 1$ for $n = 255$.

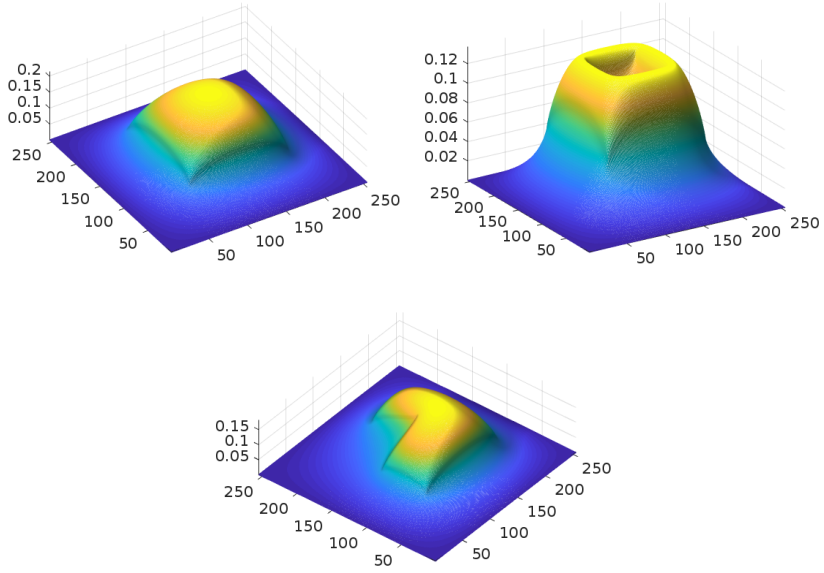


Figure 5.3: Solutions $\mathbf{u}^*$ for above right-hand sides with $\alpha = 1/2$ for $n = 255$.

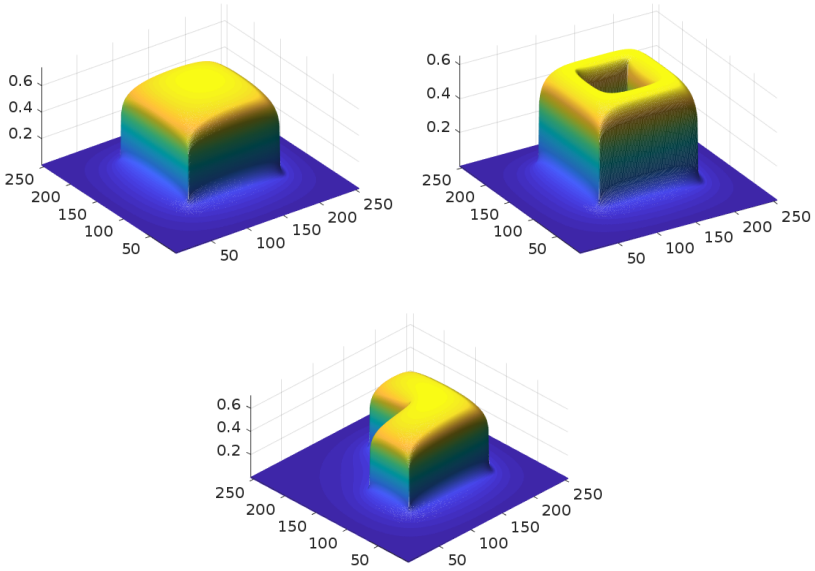


Figure 5.4: Solutions $\mathbf{u}^*$ for above right-hand sides with $\alpha = 1/10$ for $n = 255$.

One observes the nonlocal features of the elliptic resolvent $\mathcal{A}_h^{-1}$ and highly localized action of the operator $\mathcal{A}_h^{-1/10}$.

5.6.1 Numerical Tests for the 2D Case

Figure 5.5, left, represents the singular values of the matrix G_1, with entries given by (5.29) for different univariate grid size $n = 255$, 511, and 1023 and fixed $\alpha = 1$ (Laplacian inverse). Figure 5.5, right, shows the decay of respective singular values for G_1 with fixed univariate grid size $n = 511$ and for different $\alpha = 1, \frac{1}{2}, \frac{1}{10}$.

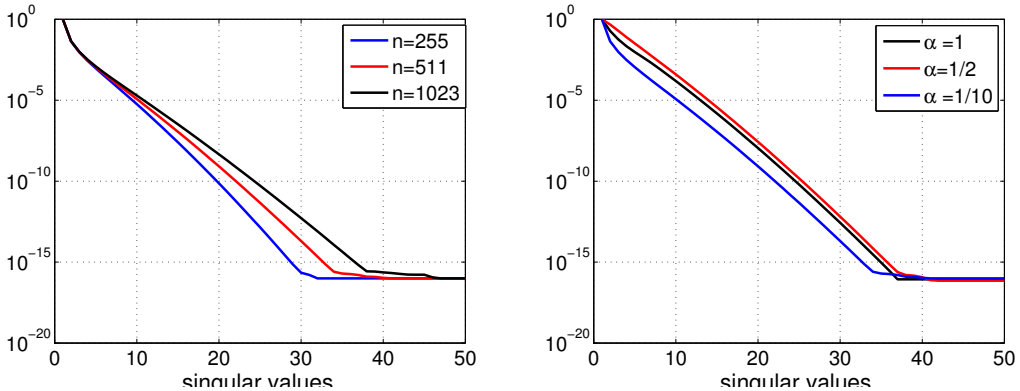


Figure 5.5: Decay of singular values for G_1 with $\alpha = 1$ vs. n (left); singular values for G_1 vs. $\alpha > 0$ with fixed $n = 511$ (right).

Figure 5.6 demonstrates the behaviour of singular values for matrices G_2 and G_3, with entries corresponding to (5.30) and (5.31), respectively, vs. $\alpha = 1, \frac{1}{2}, \frac{1}{10}$ with fixed univariate grids size $n = 511$. In all cases, we observe exponentially fast decay of the singular values, which means there exists an accurate low Kronecker-rank approximation of the matrix functions A_1, A_2 and A_3 (see equations (5.23)– (5.25)), including fractional powers of the elliptic operator.

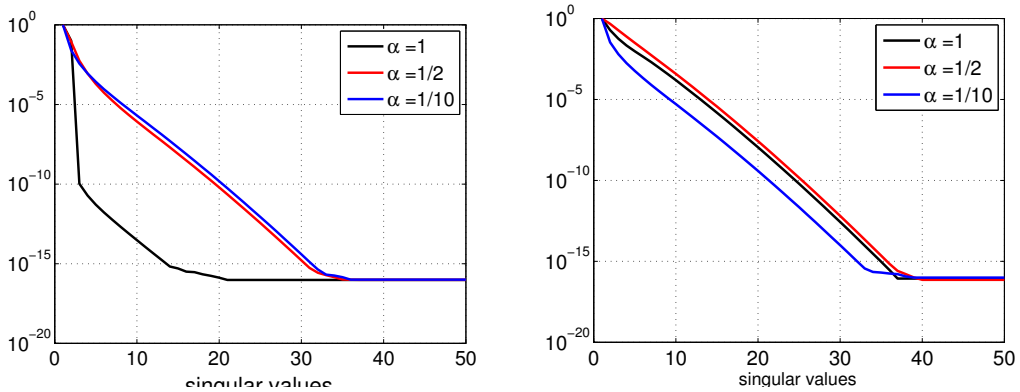


Figure 5.6: Decay of singular values of G_2 (left) and G_3 (right) vs. $\alpha = 1, 1/2, 1/10$ for $n = 511$.

Decay of the error for the optimal control obtained as the solution of equation (5.26) with rank-R approximation of the solution operator A_3 is shown in Figure 5.7.

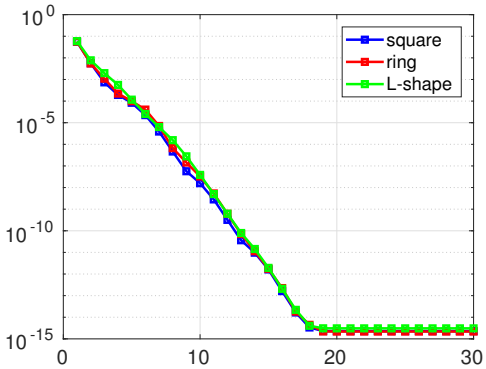


Figure 5.7: Decay of the error for the optimal control vs. truncation rank parameter.

As we have shown theoretically in Section 5.3, a single PCG iteration has a complexity, which is slightly higher than linear in the univariate grid size n. Figure 5.8 shows that the CPU times show the expected behavior. Thus, with Figure 5.8 and Tables 5.1 and 5.2, the overall cost of the algorithm is almost linear in the univariate grid size n for the problem discretized on an $n \times n$ two-dimensional Cartesian grid.

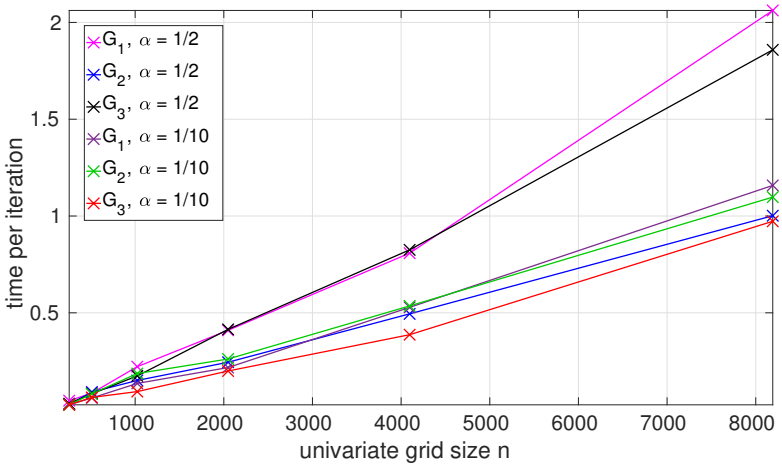


Figure 5.8: CPU times (sec) vs. univariate grid size n for a single iteration of Algorithm 5.1 for a 2D problem, for different fractional operators and fixed preconditioner rank $r = 5$.

	g_1				g_4				g_3			
r \ n	256	512	1024	2048	256	512	1024	2048	256	512	1024	2048
1	20	24	24	29	—	—	83	80	20	24	24	19
2	—	—	3	3	73	—	38	36	—	—	3	3
3	7	9	10	14	99	—	17	16	7	9	10	14
4	5	6	6	9	31	—	3	3	5	6	6	9
5	4	4	4	5	11	—	5	5	4	4	4	5
6	3	3	3	4	6	13	2	2	3	3	3	4
7	3	3	3	3	4	7	6	4	3	3	3	3
8	2	2	2	2	3	5	4	2	2	2	2	2
9	2	2	2	2	3	4	3	4	2	2	2	2
10	2	2	2	2	3	3	2	3	2	2	2	2

Table 5.1: PCG iteration counts for convergence to a relative residual of 10^{-6} for the equations (5.43)–(5.45) for a 2D fractional Laplacian with $\alpha = 1/2$ vs. the univariate grid size n and separation rank r.

We also test the properties of the low-rank discrete operator as a preconditioner. This means, we solve the equations in $\mathbb{R}^d$, $d = 2, 3$,

$$A^\alpha \mathbf{x} = \mathbf{b}, \tag{5.43}$$

$$\big(I + A^{2\alpha}\big)\mathbf{x} = \mathbf{b}, \tag{5.44}$$

$$\big(A^\alpha + A^{-\alpha}\big)\mathbf{x} = \mathbf{b}, \tag{5.45}$$

with a preconditioned conjugate gradient scheme as presented in Algorithm 5.1, using a low-rank direct solver as a preconditioner discussed above. We simplify the notation by $A = \mathcal{A}_h$.

	g_1				g_4				g_3			
r \ n	256	512	1024	2048	256	512	1024	2048	256	512	1024	2048
1	9	9	10	11	11	13	14	16	7	7	8	9
2	6	4	7	8	7	8	8	9	5	5	6	6
3	4	5	5	6	5	5	6	7	4	4	5	5
4	4	4	4	5	4	4	4	5	3	4	4	4
5	3	3	4	4	3	4	4	4	3	3	3	4
6	3	3	3	4	3	3	3	4	2	3	3	3
7	2	3	3	3	2	3	3	3	2	2	3	3
8	2	2	2	3	2	2	2	3	2	2	2	3
9	2	2	2	2	2	2	2	2	2	2	2	3
10	2	2	2	2	2	2	2	2	2	2	2	2

Table 5.2: PCG iteration counts for convergence to a relative residual of 10^{-6} for the equations (5.43)–(5.45) for a 2D fractional Laplacian with $\alpha = 1/10$ vs. the univariate grid size n and separation rank r.

In numerical tests we solve the equations (5.43)–(5.45) on a grid of size n, using a rank-r preconditioner. Tables 5.1 and 5.2 show the number of CG iteration counts for convergence to a relative residual of 10^{-6} of (5.43)–(5.45) with $\alpha = 1/2$ and $\alpha = 1/10$, respectively. The dash '—' indicates failure to converge to converge in 100 iterations.

As can be seen in Tables 5.1 and 5.2, we achieve almost grid-independent preconditioning; the iteration counts only grow logarithmically with the number of grid points, as can be expected from the theoretical reasoning. As can be seen in Table 5.1, the ranks of the preconditioner should be chosen sufficiently large to ensure reliability. In the cases tested here, $r = 6$ is sufficient to achieve reliable preconditioning even in the most difficult case of equation (5.44) with $\alpha = 1/2$.

5.6.2 Numerical Tests for the 3D Case

In the following examples we solve the problems governed by the 3D operators in (5.23)–(5.25), with a 3D fractional Laplacian with $\alpha = 1, \frac{1}{2}, \frac{1}{10}$. The rank-structured approximation to the above fractional operators is performed by using the multigrid Tucker decomposition of the 3D tensors $\mathbf{G}_k$, $k = 1, 2, 3, 4$, described by (5.34)–(5.36), and the consequent Tucker-to-canonical decomposition of the Tucker core tensor thus obtaining a canonical tensor with a smaller rank. The rank truncation procedure in the PCG Algorithm 5.1 is performed by using the RHOSVD tensor approximation and its consequent transform to the canonical format, see Section 5.4.

Figures 5.9–5.11 demonstrate the exponential convergence of the approximation error with respect to the Tucker rank for operators given by (5.43)–(5.45).

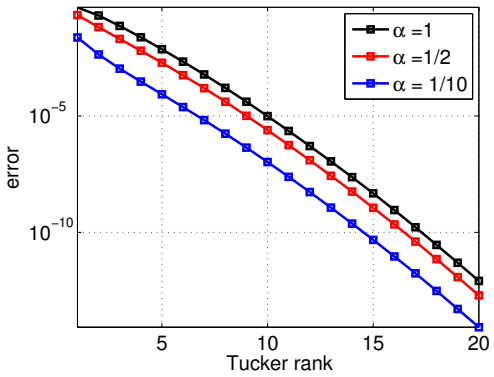
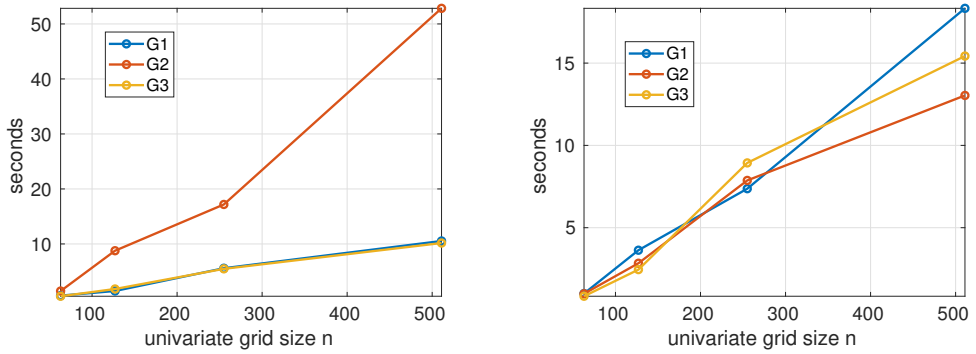


Figure 5.9: Tucker tensor approximation of $\mathbf{G}_1$ vs. rank parameter for $\alpha = 1, 1/2, 1/10$.



$\alpha = 1/2,\ r = 4$ $\qquad$ $\alpha = 1/10,\ r = 7$

Figure 5.10: CPU times (in seconds) vs. grid size n of a single iteration of Algorithm 5.1 for a 3D problem, for different fractional operators and fixed preconditioner rank r.

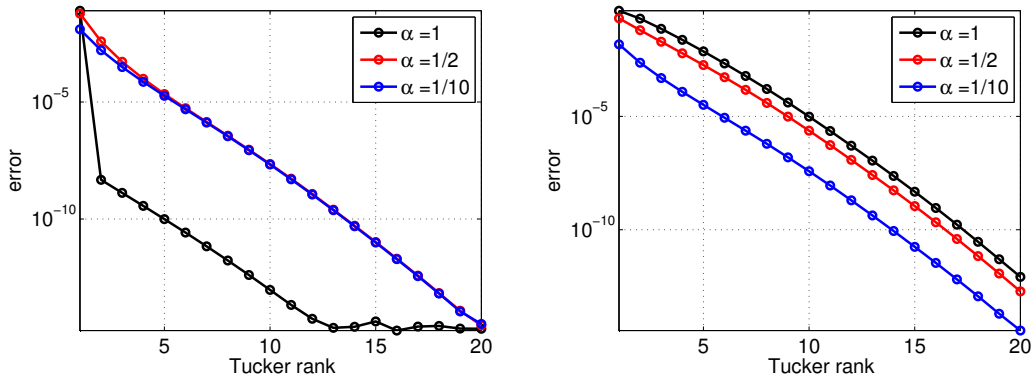


Figure 5.11: Tucker tensor approximation of $\mathbf{G}_2$ and $\mathbf{G}_3$ vs. rank parameter for $\alpha = 1, 1/2, 1/10$.

We solve the equations (5.43)–(5.45) using $n \times n \times n$ three-dimensional Cartesian grids with the univariate grid size n, using a rank-r preconditioner. Tables 5.3 and 5.4 show the number of CG iteration counts for convergence to a relative residual of 10^{-6} of (5.43)–(5.45) with $\alpha = 1/2$ and $\alpha = 1/10$, respectively.

Similarly to the previous subsection, we see that the low-rank approximation gives us an approximately grid-independent preconditioner. In the cases tested here, $r = 6$ is sufficient to achieve reliable preconditioning even in the most difficult case of equation (5.44) with $\alpha = 1/2$.

	g_1				g_4				g_3			
r \ n	64	128	256	512	64	128	256	512	64	128	256	512
4	1	2	1	1	1	6	1	2	1	2	1	1
5	1	1	1	2	1	1	8	4	1	1	1	2
6	1	1	1	1	2	2	1	1	1	1	1	1
7	1	3	1	2	1	1	5	4	1	2	1	2
8	1	1	1	1	1	1	1	1	1	1	1	1
9	1	1	1	2	1	6	5	4	1	1	1	2
10	1	1	1	1	1	6	1	1	1	1	1	1

Table 5.3: PCG iteration counts for convergence to a relative residual of 10^{-6} for the equations (5.43) - (5.45) for a 3D fractional Laplacian with $\alpha = 1/2$. Here n is the univariate grid size, r is the separation rank.

	g_1				g_4				g_3			
r \ n	64	128	256	512	64	128	256	512	64	128	256	512
4	2	1	9	20	2	1	10	17	1	1	9	18
5	1	1	1	1	1	1	1	1	1	2	1	13
6	1	1	1	2	1	1	1	2	1	1	1	7
7	1	1	1	2	1	1	1	2	1	1	2	1
8	1	1	1	1	1	1	1	1	1	1	1	2
9	1	1	1	1	1	1	1	1	1	1	1	1
10	1	1	1	2	1	1	1	2	1	1	1	1

Table 5.4: CG iteration counts for convergence to a relative residual of 10^{-6} for the equations (5.43)-(5.45) for a 3D fractional Laplacian with $\alpha = 1/10$. Here n is the univariate grid size, r is the separation rank.

Our numerical test indicates that all three matrices A_1, A_2 and A_3, as well as the corresponding three-tensors have ε-rank approximation such that the rank parameter depends logarithmically on ε, i. e. $r = O(|\log \varepsilon|)$, that means that the low rank representation of the design function y_Ω ensures the low-rank representation of both optimal control and optimal state variable.

We show as well that, using rank-structured tensor methods for the numerical solution of this optimization problem using the operators of type A_1, A_2 and A_3 can be imple-

mented at low cost that scales almost linearly in the univariate grid size, $O(n \log n)$, see Figure 5.10.

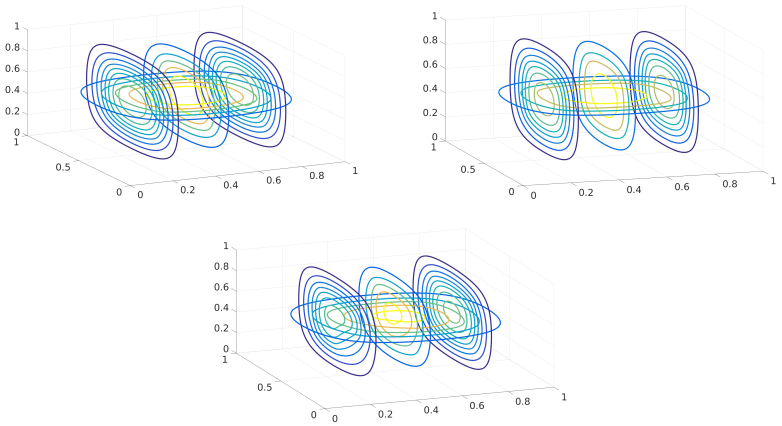


Figure 5.12: Solutions of the equation with 3D right-hand sides (analogous to Figure 5.1) with $\alpha = 1$ for $n = 255$.

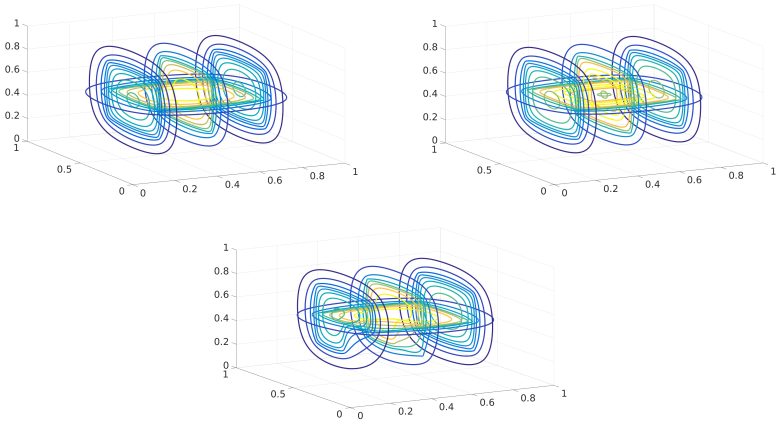


Figure 5.13: Solutions of the equation with 3D right-hand sides (analogous to Figure 5.1) with $\alpha = 1/2$ for $n = 255$.

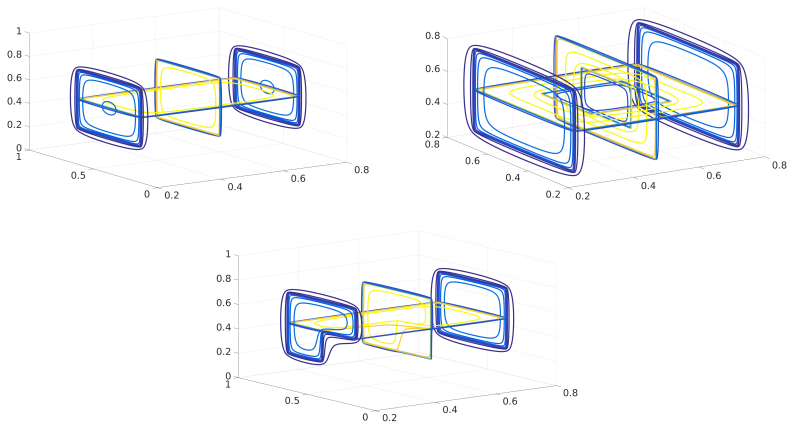


Figure 5.14: Solutions of the equation with 3D right-hand sides (analogous to Figure 5.1) with $\alpha = 1/10$ for $n = 255$.

6 Conclusion

In this thesis, we have applied tensor numerical methods in two different areas. The first application concerns tensor completion in data science.

We have derived the Riemannian Hessian for functions on the manifold of tensors of fixed multilinear rank in Tucker format. We have shown that it can be used to construct a rapidly and robustly converging trust-region scheme for tensor completion. The tensor recovery quality compares favourably with state-of-the-art methods. Furthermore, this is the first theoretical result on the second-order properties of the given manifold; we believe this to be useful for an improved understanding of the underlying geometry. Our numerical results also indicate that Riemannian optimization is a suitable technique for the recovery of missing entries from multilinear survey data with low-rank structure. We believe that this aspect merits further exploration; a comparison of Riemannian techniques with standard imputation methods from statistics [67] may reveal opportunities and limitations of this approach. For this, a better understanding of the sensitivity of the Tucker decomposition to perturbations is required.

Riemannian preconditioning techniques from recent literature [71] may be used to speed up convergence for ill-conditioned problems; an application to the geometry used here would be required. Another well-known way to obtain superlinear convergence is a Riemannian BFGS method. In recent research, several schemes have been proposed, generalizing this standard method from Euclidean optimization to the Riemannian case; see [43, Subsection 5.2] for an application to the manifold of matrices of fixed rank. In this thesis, we provide a novel vector transport for the fixed-rank tensor manifold, which may be used to generalize this BFGS method to tensors. Extending this idea to tensors merits some examination. For high-dimensional applications with $d \gg 3$, hierarchical tensor formats [88, 31] are crucial; see [20] for a Riemannian optimization approach.

Our second application lies in scientific computing, namely in the optimal control of a fractional partial differential equation.

We have introduced and analyzed a new approach for the optimal control of a fractional Laplacian equation using tensor numerical methods. The fractional Laplacian is diagonalized in the FFT basis on a tensor grid, and a low-rank approximation to the core diagonal tensor is computed. We present a novel rank-structured representation of functions of the fractional elliptic operator based on sinc-approximation method applied to the core tensor. This representation exhibits exponential decay of the approximation error in the rank parameter. These results apply to the fractional Laplacian itself, as well as to the solution operators of a fractional control problem, resulting from first-order necessary conditions. Due to the separation of the spatial variables, the application of the arising matrix-valued functions of a fractional Laplacian to a given rank-structured vector has a complexity which is nearly linear (linear-logarithmic) in the univariate grid

size, independently of the spatial dimension of the problem.

The PCG algorithm for solving the equation for control function with adaptive rank truncation is implemented. In the 3D case, the rank truncation is based on the RHOSVD-Tucker approximation and its transform to the low-rank canonical form. Our numerical study illustrates the exponential decay of the approximation error of the canonical tensor decompositions of the target tensors in the rank parameter, and indicates almost linear complexity scaling of the rank-truncated PCG solver in the univariate grid size for 3D problems. The low-rank preconditioner provides a uniform convergence rate in the grid size and other model parameters.

The approach can be applied to generalized Laplacian-type control operators and to the case of fractional elliptic operators with variable coefficients. Moreover, further reduction of the numerical complexity to the logarithmic scale can be achieved by using the quantized-TT (QTT) representation of all discrete functions and operators involved (see [49]).

Bibliography

[1] P.-A. Absil, C. G. Baker, and K. A. Gallivan. Trust-region methods on Riemannian manifolds. *Found. Comput. Math.*, 7(3):303–330, 2007.

[2] P.-A. Absil, R. Mahoney, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, 2008.

[3] P.-A. Absil, R. Mahony, and J. Trumpf. An extrinsic look at the Riemannian Hessian. In F. Nielsen and F. Barbaresco, editors, *Geometric Science of Information*, number 1, pages 361–368, 2013.

[4] H. Antil and E. Otárola. A FEM for an optimal control problem of fractional powers of elliptic operators. *SIAM J. Control Optim.*, 53(6):432–456, 2015.

[5] B. W. Bader and T. G. Kolda. Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. *ACM Trans. Math. Softw.*, 32(4):635–653, 2011.

[6] B. W. Bader, T. G. Kolda, et al. MATLAB Tensor Toolbox Version 2.6. Available online, 2015.

[7] R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, 1957.

[8] J. M. F. ten Berge. Kruskal's polynomial for $2 \times 2 \times 2$ arrays and a generalization to $2 \times n \times n$ arrays. *Psychometrika*, 56(4):631–636, 1991.

[9] A. Borzì and V. Schulz. Multigrid methods for PDE optimization. *SIAM Rev.*, 51(2):361–395, 2009.

[10] A. Borzì and V. H. Schulz. *Computational Optimization of Systems Governed by Partial Differential Equations*. SIAM, Philadelphia, 2012.

[11] N. Boumal. Riemannian trust regions with finite-difference Hessian approximations are globally convergent. In F. Nielsen and F. Barbaresco, editors, *Geometric Science of Information*, number 2, pages 467–475, 2015.

[12] N. Boumal and P.-A. Absil. RTRMC: a Riemannian trust-region method for low-rank matrix completion. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, number 24, pages 406–414, 2011.

[13] N. Boumal, B. Mishra, P.-A. Absil, R. Sepulchre, et al. Manopt, a Matlab toolbox for optimization on manifolds. *J. Mach. Learn. Res.*, 15(1):1455–1459, 2014.

[14] P. Breiding and N. Vannieuwenhoven. A Riemannian trust region method for the canonical tensor rank approximation problem. *SIAM J. Optim.*, 28(3):2435–2465, 2018.

[15] L. Caffarelli and L. Silvestre. An extension problem related to the fractional Laplacian. *Comm. Partial Differential Equations*, 32(8):1245–1260, 2007.

[16] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Found. Comput. Math.*, 9(6):717–772, 2009.

[17] 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(3):283–319, 1970.

[18] A. Cichocki and S. Amari. *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications*. Wiley, Chichester, 2002.

[19] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*. SIAM, Philadelphia, 2000.

[20] C. Da Silva and F. J. Herrmann. Optimization on the hierarchical Tucker manifold – applications to tensor completion. *Linear Algebra Appl.*, 481:131–173, 2015.

[21] S. Dolgov, J. W. Pearson, D. V. Savostyanov, and M. Stoll. Fast tensor product solvers for optimization problems with fractional differential equations as constraints. *Appl. Math. Comput.*, 273:604–623, 2016.

[22] B. Duan, R. Lazarov, and J. E. Pasciak. Numerical approximation of fractional powers of elliptic operators. arXiv preprint, 2018. arXiv:1803.10055.

[23] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[24] L. Eldén and B. Savas. A Newton–Grassmann method for computing the best multilinear rank-$(r_1,\ r_2,\ r_3)$ approximation of a tensor. *SIAM. J. Matrix Anal. Appl.*, 31(2):248–271, 2009.

[25] D. H. Foster, S. M. C. Nacimento, and K. Amano. Information limits on neural identification of colored surfaces in natural scenes. *Vis. Neurosci.*, 21(3):331–336, 2004.

[26] S. Gandy, B. Recht, and I. Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Probl.*, 27(2):025010, 2011.

[27] I. P. Gavrilyuk, W. Hackbusch, and B. N. Khoromskij. Data-sparse approximation to the operator-valued functions of elliptic operator. *Computing*, 73:1297–1324, 2003.

[28] I. P. Gavrilyuk, W. Hackbusch, and B. N. Khoromskij. Hierarchical tensor-product approximation to the inverse and related operators for high-dimensional elliptic problems. *Computing*, 74(2):131–157, 2005.

[29] G. H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM J. Numer. Anal.*, 10(2):413–432, 1973.

[30] L. Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM. J. Matrix Anal. Appl.*, 31(4):2029–2054, 2010.

[31] L. Grasedyck, D. Kressner, and C. Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitt.*, 36(1):53–78, 2013.

[32] W. Hackbusch. *Tensor Spaces and Numerical Tensor Calculus.* Springer, Berlin, 2012.

[33] W. Hackbusch and S. Kühn. A new scheme for the tensor representation. *J. Fourier Anal. Appl.*, 15:706–722, 2009.

[34] N. Hale, N. J. Higham, and L. N. Trefethen. Computing a^α, $\log(a)$, and related matrix functions by contour integrals. *SIAM J. Numer. Anal*, 46(2):2505–2523, 2008.

[35] S. Harizanov, R. Lazarov, S. Margenov, P. Marinov, and Y. Vutov. Optimal solvers for linear systems with fractional powers of sparse SPD matrices. *Numer. Linear Algebra Appl.*, 25(5):e2167, 2018.

[36] R. A. Harshman. Foundations of the PARAFAC procedure: models and conditions for an "explanatory" multimodal factor analysis. In *UCLA Working Papers in Phonetics*, volume 16, pages 1–84. 1970. Available at `http://www.psychology.uwo.ca/faculty/harshman/wpppfac0.pdf`.

[37] G. Heidel, V. Khoromskaia, B. N. Khoromskij, and V. Schulz. Tensor approach to optimal control problems with fractional d-dimensional elliptic operator in constraints. arXiv preprint, 2018. arXiv:1809.01971, submitted.

[38] G. Heidel and V. Schulz. A Riemannian trust-region method for low-rank tensor completion. *Numer. Linear Algebra Appl.*, 25(6):e2175, 2018.

[39] R. Herzog and K. Kunisch. Algorithms for PDE-constrained optimization. *Computing*, 33(2):163–176, 2010.

[40] N. J. Higham. *Functions of Matrices: Theory and Computation.* SIAM, Philadelphia, 2008.

[41] F. L. Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Stud. Appl. Math.*, 6(1–4):164–189, 1927.

[42] F. L. Hitchcock. Multiple invariants and generalized rank of a p-way matrix or tensor. *Stud. Appl. Math.*, 7(1–4):39–79, 1928.

[43] W. Huang, P.-A. Absil, and K. A. Gallivan. Intrinsic representation of tangent vectors and vector transports on matrix manifolds. *Numer. Math.*, 136(2):523–543, 2017.

[44] J. Håstad. Tensor rank is NP-complete. *J. Algorithms*, 11(4):644–654, 1990.

[45] M. Ishteva, P.-A. Absil, S. van Huffel, and L. de Lathauwer. Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme. *SIAM J. Matrix Anal. Appl.*, 32(1):115–135, 2011.

[46] H. Kasai and B. Mishra. Low-rank tensor completion: a Riemannian manifold preconditioning approach. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of Machine Learning Research*, number 48, pages 1012–1021, 2016.

[47] V. Khoromskaia and B. N. Khoromskij. *Tensor Numerical Methods in Quantum Chemistry*. De Gruyter, Berlin, 2018.

[48] B. N. Khoromskij. Structured rank-$(R_1, \cdots, R_D)$ decomposition of function-related tensors in $\mathbb{R}^D$. *Comput. Methods Appl. Math.*, 6(2):194–220, 2006.

[49] B. N. Khoromskij. $O(d \log N)$-Quantics approximation of N-d tensors in high-dimensional numerical modeling. *Constr. Approx.*, 34(2):257–280, 2011.

[50] B. N. Khoromskij. *Tensor Numerical Methods in Scientific Computing*. De Gruyter, Berlin, 2018.

[51] B. N. Khoromskij and V. Khoromskaia. Low rank Tucker-type tensor approximation to classical potentials. *Open Math.*, 5(3):523–550, 2007.

[52] B. N. Khoromskij and V. Khoromskaia. Multigrid accelerated tensor approximation of function related multidimensional arrays. *SIAM J. Sci. Comput.*, 31(4):3002–3026, 2009.

[53] H. A. L. Kiers. Towards a standardized notation and terminology in multiway analysis. *J. Chemometrics*, 14(3):105–122, 2000.

[54] O. Koch and C. Lubich. Dynamical tensor approximation. *SIAM J. Matrix Anal. Appl.*, 31(5):2360–2375, 2007.

[55] T. G. Kolda. Orthogonal tensor decompositions. *SIAM J. Matrix Anal. Appl.*, 23(1):243–255, 2001.

[56] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):131–173, 2009.

[57] D. Kressner, M. Steinlechner, and B. Vandereycken. Low-rank tensor completion by Riemannian optimization. *BIT*, 54(2):447–468, 2014.

[58] D. Kressner, M. Steinlechner, and B. Vandereycken. Preconditioned low-rank Riemannian optimization for linear systems with tensor product structure. *SIAM J. Sci. Comp.*, 38(4):A2018–A2044, 2016.

[59] D. Kressner and C. Tobler. Krylov subspace methods for linear systems with tensor product structure. *SIAM. J. Matrix Anal. Appl.*, 31(4):1688–1714, 2011.

[60] P. M. Kroonenberg. Information on Bus' learning-to-read data. `http://www.leidenuniv.nl/fsw/three-mode/data/businfo.htm`. Accessed: 6 January 2019.

[61] P. M. Kroonenberg. *Three-Mode Principal Component Analysis: Theory and Applications*. PhD thesis, Universiteit Leiden, 1983.

[62] J. B. Kruskal. Rank, decomposition, and uniqueness for 3-way and n-way arrays. In R. Coppi and S. Bolasco, editors, *Multiway Data Analysis*, pages 7–18. North-Holland, Amsterdam, 1989.

[63] M. Kwaśnicki. Ten equivalent definitions of the fractional Laplace operator. *Fract. Calc. Appl. Anal.*., 20(1):7–51, 2017.

[64] L. De Lathauwer, B. de Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.

[65] L. de Lathauwer, B. de Moor, and J. Vandewalle. On the best rank-1 and rank-$(R_1, R_2, \ldots, R_N)$ approximation of higher-order tensors. *SIAM J. Matrix Anal. Appl.*, 21(4):1324–1342, 2000.

[66] A. Lischke, G. Pang, M. Gulian, F. Song, C. Glusa, X. Zheng, Z. Mao, W. Cai, M. M. Meerschaert, M. Ainsworth, and G. E. Karniadakis. What is the fractional Laplacian? arXiv preprint, 2018. arXiv:1801.09767.

[67] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. Wiley, New York, 2014.

[68] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):208–220, 2013.

[69] H. Lütkepohl. *Handbook of Matrices*. Wiley, Chichester, 1997.

[70] B. Mishra, G. Meyer, S. Bonnabel, and R. Sepulchre. Fixed-rank matrix factorizations and Riemannian low-rank optimization. *Comput. Stat.*, 29(3):591–621, 2014.

[71] B. Mishra and R. Sepulchre. Riemannian preconditioning. *SIAM J. Optim.*, 26(1):635–660, 2016.

[72] T. T. Ngo and Y. Saad. Scaled gradients on Grassmann manifolds for matrix completion. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, number 25, pages 1412–1420, 2012.

[73] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2006.

[74] I. V. Oseledets. Tensor-train decomposition. *SIAM J. Sci. Comput.*, 33(5):2295–2317, 2011.

[75] I. V. Oseledets and S. V. Dolgov. Solution of linear systems and matrix inversion in the TT-format. *SIAM J. Sci. Comput.*, 34(5):A2718–A2739, 2012.

[76] P. Paatero. Construction and analysis of degenerate PARAFAC models. *J. Chemometrics*, 14(3):285–299, 2000.

[77] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1992.

[78] M. Signoretto, L. de Lathauwer, and J. A. K. Suykens. Nuclear norms for tensors and their use for convex multilinear estimation. Technical report, 2010.

[79] M. Signoretto, R. Van de Plas, B. De Moor, and J. A. K. Suykens. Tensor versus matrix completion: a comparison with application to spectral data. *IEEE Signal Process. Lett.*, 18(7):403–406, 2011.

[80] V. de Silva and L.-H. Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM. J. Matrix Anal. Appl.*, 30(3):1084–1127, 2008.

[81] Q. Song, H. Ge, J. Caverlee, and X. Hu. Tensor completion algorithms in big data analytics. arXiv preprint, 2017. arXiv:1711.10105.

[82] M. M. Steinlechner. *Riemannian Optimization for Solving High-Dimensional Problems with Low-Rank Tensor Structure*. PhD thesis, École polytechnique fédérale de Lausanne, 2016.

[83] G. Tomasi and R. Bro. A comparison of algorithms for fitting the PARAFAC model. *Comput. Statist. Data Anal.*, 50(7):1700–1734, 2006.

[84] F. Tröltzsch. *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*. American Mathematical Society, Providence, 2010.

[85] L. W. Tu. *An Introduction to Manifolds*. Springer, New York, 2011.

[86] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[87] C. Udriște. *Convex Functions and Optimization Methods on Riemannian Manifolds*. Kluwer Academic Publishers, Dordrecht, 1994.

[88] A. Uschmajew and B. Vandereycken. The geometry of algorithms using hierarchical tensors. *Linear Algebra Appl.*, 439(1):133–166, 2013.

[89] P. N. Vabishchevich. Numerically solving an equation for fractional powers of elliptic operators. *J. Comput. Phys.*, 5:289–302, 2015.

[90] P. N. Vabishchevich. Numerical solution of time-dependent problems with fractional power elliptic operator. *Comput. Methods Appl. Math.*, 18(1):111–128, 2018.

[91] B. Vandereycken. Low-rank matrix completion by Riemannian optimization. *SIAM J. Optim.*, 23(2):1214–1236, 2013.