Low Rank Tensor Decompositions for High Dimensional Data Approximation, Recovery and Prediction

vorgelegt von
Master of Science

Alexander Sebastian Johannes Wolf Wolf

geboren in Göttingen

von der Fakultät II – Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften

Dr. rer. nat.

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Peter Bank
Gutachter:	Prof. Dr. Reinhold Schneider
Gutachterin:	Prof. Dr. Gitta Kutyniok
Gutachter:	Prof. Dr. Lars Grasedyck

Tag der wissenschaftlichen Aussprache: 5. Dezember 2018

Berlin 2019

Acknowledgements

Let me express my deep gratitude to all the wonderful people who accompanied and supported me in the preparation of this thesis: This would not have been possible without you. In particular, I would like to express my special thanks to:

My supervisor Prof. Dr. Reinhold Schneider, for introducing me to this exciting field of multilinear algebra and for giving me the opportunity to become part of this research myself in his group at TU Berlin. Reinhold, thank you for being such a friendly mentor in this exciting time.

Prof. Dr. Kutyniok and Prof. Dr. Grasedyck, who kindly accepted to act as reviewers for this thesis. Thank you very much for this commitment.

My dear friend, colleague and former flat-mate Benjamin Huber, for countless great and inspiring discussions, late nights of programming and finally, for many useful remarks on the manuscript of this work. Ben, thank you so much for all the projects we worked on together and the great time we had doing so.

My other colleagues and collaborators Max Pfeffer, Martin Eigel, Philipp Trunschke and Benjamin Kutschan. Thank you for the inspiring discussions and the nice and entertaining coffee breaks we had. I really enjoyed working with you.

Alexandra Schulte, for keeping the research group well organized and the bureaucratic burden to a minimum for everyone. Alex, I think we would all agree that without you everything would break down. Thank you for preventing that every day anew.

My lovely writing mates Ben, Elias, Insa, Isa and Verena. It was always a pleasure to share a desk with you and I wish all of you the very best for your respective projects. Special thanks to you, Elias, for the final proofreading of the manuscript of this work.

My family, who always supported me and more than once provided me the retreat I needed to finish this thesis.

Marion, for her endurance and her support.

Finally, I want to thank the Einstein Center for Mathematics Berlin for their financial support of the Matheon projects SE10 and CH19.

Abstract

In this thesis, we examine different approaches for efficient high dimensional data acquisition and reconstruction using low rank tensor decomposition techniques. High dimensional here refers to the *order* of the ambient tensor space in which the data is contained. Examples of such data include tomographic videos, solutions to parametric differential equations and quantum states of many particle systems. The major problem faced in any such high dimensional setting is the exponential scaling of the tensor space dimension with respect to the order, often referred to as the *curse of dimensionality.* A possible remedy are low rank tensor decomposition techniques, which allow for the storage and manipulation of a rich set of tensors in a data-sparse way, often avoiding the exponential scaling with respect to the order. Following the ideas of compressive sensing, we examine methods to exploit these very efficient representations to efficiently acquire high dimensional data from measurements. Our first approach uses randomized methods to construct a low rank representation using repeated random evaluations of the data. The second approach works non-intrusively and aims to reconstruct a complete low rank representation from different types of given (incomplete) measurements. The latter approach is known as the tensor recovery problem and contains as a special case the popular tensor completion problem. Finally, we employ tensor recovery techniques in a statistical learning setting to obtain low rank approximations of the complete solutions of parametric differential equations. This parametric representation in particular allows to predict the solution for *any* parameter combinations, even if those were not measured.

Contents

1 Introduction

One of the greatest challenges in computational mathematics today is undoubtedly the efficient handling of large data sets. This is in part due to the impressive growth of available and generated data. For example, Hilbert and López [1] show that from 1986 to 2007, the world's technological per-capita capacity to communicate and store information has increased by an impressive compound annual growth rate of 28% and 23%, respectively. Additionally, the capabilities to gather raw data continuously increase, as sensors, such as RFID readers, cameras, microphones, thermostats, etc., become more precise and less expensive [2]. At the same time, models and simulations in the natural sciences, engineering and other applications require this data in order to meet the ever increasing demands for accuracy and reliability [3].

When referring to the "efficient handling" of data, we usually have three separate tasks in mind: *data acquisition*, *data storage* and *data usage*. Efficient data acquisition refers to the minimization of the measurement process required to obtain the desired data. Efficient data storage in turn refers to data-sparse means to store such information, achieved through compression of the raw data. Efficient use of data is usually understood as the computationally thrifty extraction of reliable information from the data. While conceptually separate, all three objectives are interconnected and can sometimes even be achieved in one process. For example, a data-sparse representation, e.g. of sparse vectors or low rank matrices, often allows computationally inexpensive access to information encoded in those objects. Especially the fusion of the measurement and compression process sparked a lot of interest in recent years and led to the development of the thriving field of *compressive sensing*. Here, the fundamental idea is that if some information is compressible, i.e. can be represented by a small number of parameters, then it should also be possible to completely determine it using a similarly small number of measurements. In other words, compression and measurement may happen simultaneously, hence the name *compressive sensing*. The initial results by Donoho [4], as well as Candès et al. [5], were for signal processing and showed that sparse signal vectors can be reconstructed exactly from considerably fewer measurements than their dimension would suggest, provided that it is known that most of the entries are zero. Note that in particular one does not require the knowledge which exact entries are zero, but merely that most are. The results rapidly found application in various fields, such as medical imaging [6–8] machine learning [9–11] and quantum physics [12], to name a few. An extensive overview of fundamental papers, extensions and applications can be found on the website of Rice University [13].

In this work, we are concerned with different ways to apply the idea of compressive sensing to high dimensional data, i.e. data which can be represented as a higher order tensor. A fundamental difficulty in handling such high dimensional data is the so-called *curse of dimensionality*, which describes the exponential scaling of the storage and computational complexity of most operations with respect to the order. For large orders, this usually renders the direct storage of such tensors unfeasible, making the use of compressed representations mandatory at all times. In order to determine such a tensor from measurements, it is clear that fusing the acquisition and compression processes is the only viable choice. A very successful class of data-sparse representations of higher order tensors are the so-called *low rank tensor decompositions*, which generalize the singular value decomposition and thereby the concept of rank from matrices to higher order tensors and which will be the main tools of this thesis. The concept of tensor decompositions is already long known in form of the *canonical polyadic* (CP) decomposition introduced by Hitchcock [14] in 1927 and the *Tucker* decomposition first introduced by Tucker [15] in 1966. However, it was only recently that the concept experienced a renaissance through the development of the *hierarchical Tucker* (HT) decomposition by Hackbusch and Kühn [16] and Oseledets and Tyrtyshnikov [17], which for the first time combined a linear scaling in the order for most operations with a subspace based rank definition. This allowed a generalized hierarchical SVD and a manifold structure of the set of fixed rank tensors [18, 19]. As we will show in detail, these unique properties allow the use of very high order tensors in a mathematically well defined and stable framework, while remaining numerically feasible. The focus of this thesis is on a powerful special case of the HT decomposition, namely the *tensor train* (TT) decomposition introduced by Oseledets [20], which is also known in physics as *Matrix Product States* (MPS) [21, 22]. All of the mentioned formats allow for tremendous reduction in storage and computational complexity of common operations for a rich set of corresponding low rank tensors. Especially the performance of the CP, TT and HT formats is noteworthy as all those formats break the curse of dimensionality in the sense that for a fixed rank, the aforementioned complexities scale only linearly in the order, in contrast to the exponential scaling for general tensors. It has been shown that in many applications, the appearing tensors (approximately) have a low rank in one or more of these formats, allowing the use of low rank decompositions for increased efficiency. Prominent applications include (quantum) physics [21–25], computational chemistry [26–29], neuroscience [30–32], signal processing [33, 34], machine learning [35–38], computer vision [39–42] and graph analysis [43, 44], see also the surveys by Kolda and Bader [45] and Grasedyck et al. [46].

In the course of this thesis, we pursue two different approaches for the efficient acquisition of high dimensional data directly in one of the low rank formats. The first one is based upon the work of Halko et al. [47], who developed and analyzed randomized techniques for the calculation of different low rank matrix decompositions. While randomized algorithms for those tasks have been proposed many times in the literature, it was only rather recently that, thanks to the application of new insights from *random matrix theory*, these procedures could be analyzed rigorously and stochastic error bounds became available. In this work, we present generalizations of these algorithms to calculate low rank Tucker and Tensor

Train decompositions. For these randomized HOSVD and TT-SVD algorithms, we provide stochastic error bounds. The second approach is based on the results of Recht et al. [48] and Candès and Recht [49], who showed that *low rank matrices* can be reconstructed exactly from far fewer measurements than they have entries, using a low rank assumption. In this context, one measurement is abstractly defined as the knowledge of a linear superposition of entries of the matrix or, in the special case of so-called *matrix completion*, as knowledge of individual entries. Generalizing these ideas from matrices to higher order tensors using the rank definition provided by one of the tensor decomposition formats gives rise to the tensor recovery and tensor completion problems. These problems received considerable interest in the last years and substantial progress was made, with examples and results being available for tensor recovery/completion in the CP format [50–54], the Tucker format [42, 55–58] and the TT/HT format [59–67]. However, as will be shown in detail during the course of this thesis, many problems of tensor recovery are still fundamentally open and worth investigating. The contributions of this thesis are outlined in the following.

1.1 Outline

This thesis is structured in six chapters building on one another, which introduce the low rank tensor decompositions and present our results concerning the decomposition of (implicitly) given tensors, the reconstruction of tensors from incomplete measurements and finally, applications of these reconstruction techniques to uncertainty quantification. The contents of each chapter and the contributions covered therein are outlined in the following.

Chapter 2 lays the foundation for all further chapters by giving a formal definition of the tensor product, tensor spaces and higher order tensors, thereby introducing the notation used throughout this thesis. The chapter starts with a brief review of the formal tensor product of Hilbert-spaces and some fundamental properties. The main part gives a detailed review of the finite-dimensional real tensor spaces $\mathbb{R}^{n_1 \times \cdots \times n_d}$, as these are of highest importance for this thesis. This includes an introduction to the important concepts of *matricizations* and *tensor contractions*, as well as to *tensor networks* and the diagrammatic notation used to depict them.

Chapter 3 introduces the main tools of this thesis: the low rank tensor decompositions. We motivate the construction of several different formats with a review of the singular value decomposition (SVD) for matrices. Generalizing different aspects of this outstanding matrix decomposition, we derive the *canonical polyadic* (CP) format, the *Tucker* format and the *Tensor-Train* (TT) format and discuss advantages and disadvantages of each format. We show that for all three formats, a tremendous reduction in storage and computational complexity for common operations can be achieved, if the tensors have a low rank in the corresponding format. In this respect, especially the CP and the TT format shine, as they break the so-called *curse of dimensionality*: For a fixed rank, the computational complexity of most operations scale only linearly in the order, as opposed to the exponential scaling for general tensors. We also give a detailed description of the HOSVD and TT-SVD algorithms, which are the prime tools used to obtain quasi-best low rank approximations of given tensors in the Tucker and TT-format. The Tucker and TT format both use a subspace based definition, which allows a smooth manifold structure for the corresponding sets of fixed rank tensors, as we show by reference to several earlier works. In particular, for the TT-format we collect results which include a closed form for the tangent space and the associated orthogonal projection of this manifold. Finally, we provide a brief overview of more general decomposition formats, such as the hierarchical Tucker decomposition and provide a comparison between the different formats.

Chapter 4 is concerned with randomized methods for the calculation of low rank tensor decompositions. The HOSVD and TT-SVD algorithms introduced in Chapter 3 both exhibit an exponential scaling with respect to the order and cannot easily exploit a structured representation of the target tensor. As a possible remedy we propose randomized HOSVD and TT-SVD algorithms, for which we prove stochastic error bounds. It is shown that the computational complexity of the randomized TT-SVD scales only linearly in the order when applied to sparse tensors, making it a powerful tool for the decomposition of such tensors. We conduct extensive numerical experiments validating the error bounds and demonstrating the quality of the randomized TT-SVD. As an outlook to future work, we discuss beneficial applications of the randomized techniques to various algorithms and the possibility of extending the stochastic error bounds to existing algorithms, such as the alternating least squares (ALS) algorithm with random initialization.

Chapter 5 examines methods to further reduce the information required to obtain a low rank representation of a tensor. In particular, the possibility to reconstruct unknown tensors from incomplete linear measurements is considered. Two kinds of measurements are of special interest for this thesis. The first is the tensor completion setting, where single entries of the target tensor are known. The second are so-called rank-one samples, which are introduced in this thesis. Here we assume that a set of (complete) contractions of the target tensor with rank one tensors is known. In contrast to the previous chapter, we assume that these measurements can in general not be chosen freely and are not adaptive. This so-called *tensor recovery* setting is the high dimensional generalization of the popular matrix recovery problem. We provide a review of prior results for the matrix case and give a formal definition for the tensor recovery problem. Unlike the matrix recovery problem, which can often be relaxed into a convex nuclear norm minimization problem, the question for such a convex relaxation for the tensor recovery problem is still largely open. Nevertheless, we are able to provide and collect some results on the (unique) recoverability, as well as algorithms designed to solve the tensor recovery problem. Our focus is on the *(block) alternating steepest descent* (ASD) algorithm, which combines a gradient descent algorithm with a partial tangent space projection to a very efficient local optimization algorithm on the low rank TT manifold. We show that in the completion setting and for rank-one samples, this algorithm allows a very efficient implementation exhibiting a computational complexity scaling only linearly in the order. For this ASD algorithm, we furthermore propose a possible rank adaption strategy based on parallel optimizations in a shared block-TT format. In a range of benchmark problems, we examine the conditions for successful reconstruction and the performance of the ASD algorithm.

Chapter 6 explores the application of these tensor recovery techniques as a means for *uncertainty quantification* (UQ). We provide a brief introduction to the parametric UQ setting, in which one aims to incorporate and quantify random effects common in many real-world settings. Motivated by this practical application, we introduce an abstract *Variational Monte Carlo* formalism and explore some interesting connections between tensor recovery and machine learning. We provide a version of the alternating steepest descent algorithm specialized to the UQ recovery problem, which reconstructs a complete parametric representation of the solution of parametric PDEs using only the deterministic solutions obtained for a limited number of fixed parameter combinations. From this representation, we can obtain many statistical properties of the solution, such as the expectation value and the variance, but we are also able to predict the solution to new parameter combinations *not* used for the reconstruction. The practical performance of the algorithms and the quality of the reconstruction is examined in various numerical experiments.

2 Tensors and Tensor Spaces

This chapter introduces the foundations of higher order tensors, tensor spaces and the tensor product. In Section 2.1, we begin with a formal definition for the tensor product of Hilbert spaces and provide some fundamental properties. As this generally is scarcely needed in this thesis, we focus on the finite-dimensional real spaces $\mathbb{R}^{n_1\times\cdots\times n_d}$, which are described in detail in Section 2.2, where additionally much of the notation necessary for the later chapters is introduced. Finally, Section 2.3 presents the concept of tensor networks and the diagrammatic notation used to depict them.

2.1 Tensor Product of Hilbert Spaces

The definition for the tensor product of Hilbert spaces given in this section follows the respective introductions in the works of Uschmajew [68], as well as Reed and Simon [69]. For an exhaustive introduction, including the treatment of Banach tensor spaces, we refer to the monograph of Hackbusch [70], where the proofs omitted in this chapter can also be found.

For the remainder of this section, let U and V be Hilbert spaces over the same field $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$. The respective scalar products $\langle\cdot,\cdot\rangle_U$ and $\langle\cdot,\cdot\rangle_V$ are assumed to be conjugated linear in the second argument. The tensor product between elements from U and V can then be defined as follows.

Definition 2.1 (Algebraic Tensor Product)**.** For any $\boldsymbol{u} \in U$ and $\boldsymbol{v} \in V$, the conjugate bilinear form

$$\begin{aligned} \boldsymbol{u}\otimes\boldsymbol{v}\ \ &: U \times V \to \mathbb{R} \\ &(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{v}}) \mapsto \langle \boldsymbol{u}, \tilde{\boldsymbol{u}}\rangle_U \cdot \langle \boldsymbol{v}, \tilde{\boldsymbol{v}}\rangle_V\ , \end{aligned}$$

is called the tensor product of $\boldsymbol{u}$ and $\boldsymbol{v}$. Elements that can be constructed in this way are called *elementary tensors*. The linear span (using the canonical addition) of all elementary tensors

$$U \otimes_a V := \operatorname{span}\{\boldsymbol{u}\otimes\boldsymbol{v} \mid \boldsymbol{u}\in U, \boldsymbol{v}\in V\}$$

is called the algebraic tensor product of U and V.

The scalar products on U and V induce the following scalar product on the tensor product space.

Proposition 2.2 (Induced Scalar Product [69, Proposition II.4.1])**.** *For every $\mathcal{X}, \tilde{\mathcal{X}} \in U \otimes_a V$, the* induced scalar product $\langle \mathcal{X}, \tilde{\mathcal{X}} \rangle_{U \otimes V}$ *is defined via any (finite) linear representations*

$$\mathcal{X} = \sum_i \boldsymbol{u}_i \otimes \boldsymbol{v}_i \qquad \boldsymbol{u}_i \in U, \boldsymbol{v}_i \in V$$

$$\tilde{\mathcal{X}} = \sum_j \tilde{\boldsymbol{u}}_j \otimes \tilde{\boldsymbol{v}}_j \qquad \tilde{\boldsymbol{u}}_j \in U, \tilde{\boldsymbol{v}}_j \in V$$

as

$$\langle \mathcal{X}, \tilde{\mathcal{X}} \rangle_{U \otimes V} := \sum_{i,j} \langle \boldsymbol{u}_i, \tilde{\boldsymbol{u}}_j \rangle_U \cdot \langle \boldsymbol{v}_i, \tilde{\boldsymbol{v}}_j \rangle_V \ .$$

This induced scalar product *on $U \otimes_a V$ is well defined, i.e. does not depend on the linear representations chosen.*

If U and V are finite-dimensional, $U \otimes_a V$ equipped with the induced scalar product is itself a Hilbert space. However, note that in general this is not the case, because $U \otimes_a V$ might not be complete. This motivates the following definition of the (topological) tensor product, which yields a complete Hilbert space.

Definition 2.3 (Tensor Product)**.** The closure of the algebraic tensor product

$$U \otimes V := \overline{U \otimes_a V} = \overline{\operatorname{span}\{\boldsymbol{u} \otimes \boldsymbol{v} \mid \boldsymbol{u} \in U, \boldsymbol{v} \in V\}}$$

with respect to the norm $\|\cdot\|_{U \otimes V}$ defined by the induced scalar product $\langle \cdot, \cdot \rangle_{U \otimes V}$ is called the (topological) tensor product of U and V.

A beneficial property that will be important later in this thesis, namely the fact that bases and subspaces of U and V have a direct correspondence in the tensor product space $U \otimes V$, is formalized in the following propositions.

Proposition 2.4 (Tensor Basis [69, Proposition II.4.2])**.** *If $\{\boldsymbol{u}_i \mid 1 \leq i \leq \dim(U)\}$ and $\{\boldsymbol{v}_j \mid 1 \leq j \leq \dim(V)\}$ are orthonormal bases of U and V, respectively, then $\{\boldsymbol{u}_i \otimes \boldsymbol{v}_j \mid 1 \leq i \leq \dim(U),\ 1 \leq j \leq \dim(V)\}$ is an orthonormal basis of $U \otimes V$. Therefore, the dimension of $U \otimes V$ is given as $\dim(U) \cdot \dim(V)$.*

Proposition 2.5 (Tensor Subspaces [68, Satz 1.4])**.** *Let $\tilde{U} \subseteq U$ and $\tilde{V} \subseteq V$ be subspaces of U and V and $\tilde{U}^\perp$ and $\tilde{V}^\perp$ the respective orthogonal complements, then*

$$U \otimes V = \left(\tilde{U} \otimes \tilde{V}\right) \oplus \left(\tilde{U}^\perp \otimes \tilde{V}\right) \oplus \left(\tilde{U} \otimes \tilde{V}^\perp\right) \oplus \left(\tilde{U}^\perp \otimes \tilde{V}^\perp\right)$$

is an orthogonal decomposition of the space $U \otimes V$. In particular, $\tilde{U} \otimes \tilde{V}$ is a subspace of $U \otimes V$ with dimension $\dim(\tilde{U}) \cdot \dim(\tilde{V})$.

It is often of interest to regard tensor spaces resulting from multiple tensor products. For this, let $U_1, \ldots, U_d$ be Hilbert spaces over the same field $\mathbb{K}$ with scalar products $\langle \cdot, \cdot \rangle_{U_\mu}$, which are conjugated linear in the second argument. Analogously to Definition 2.1, the tensor product space of these d Hilbert spaces can be defined via the following conjugate multilinear forms.

Definition 2.6 (Multiple Algebraic Tensor Product)**.** For any $\boldsymbol{u}_1 \in U_1, \ldots, \boldsymbol{u}_d \in U_d$, the conjugate multilinear form

$$\begin{aligned} \boldsymbol{u}_1 \otimes \ldots \otimes \boldsymbol{u}_d \ &: U_1 \times \ldots \times U_d \to \mathbb{R} \\ &(\tilde{\boldsymbol{u}}_1, \ldots, \tilde{\boldsymbol{u}}_d) \mapsto \langle \boldsymbol{u}_1, \tilde{\boldsymbol{u}}_1 \rangle_{U_1} \cdot \ldots \cdot \langle \boldsymbol{u}_d, \tilde{\boldsymbol{u}}_d \rangle_{U_d} \ , \end{aligned}$$

is called the tensor product of $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d$ and elements which can be constructed in this way are called *elementary tensors.* The linear span (using the canonical addition) of all elementary tensors

$$U_1 \otimes_a \ldots \otimes_a U_d := \operatorname{span}\{\boldsymbol{u}_1 \otimes \ldots \otimes \boldsymbol{u}_d \mid \boldsymbol{u}_\mu \in U_\mu\}$$

is called the algebraic tensor product of $U_1, \ldots, U_d$. The number d of involved Hilbert spaces is referred to as the *order* of the tensor space.

All properties presented above extend to these higher order tensor spaces straightforwardly. Analogously to the binary case, the topological tensor product is defined as the closure of the algebraic tensor product with respect to the induced scalar product. Using Proposition 2.4, one can show that the tensor product is associative (up to isomorphisms) and that $(U_1 \otimes U_2) \otimes U_3 = U_1 \otimes (U_2 \otimes U_3) = U_1 \otimes U_2 \otimes U_3$ holds.[1]

The greater part of this thesis is confined to finite-dimensional, real Hilbert spaces U_μ. By fixing bases $\{\phi_{\mu,i_\mu} \mid i_\mu = 1, \ldots, n_\mu\}$, these spaces are isomorphic to some $\mathbb{R}^{n_\mu}$ and every element of the tensor space $\mathcal{X} \in U_1 \otimes \ldots \otimes U_d$ can be expressed as

$$\mathcal{X} = \sum_{i_1,\ldots,i_d} \mathcal{C}_{i_1,\ldots,i_d} \ \phi_{1,i_1} \otimes \ldots \otimes \phi_{d,i_d}$$

for a $\mathcal{C} \in \mathbb{R}^{n_1\times\ldots\times n_d}$. As $U_1 \otimes \ldots \otimes U_d$ is thereby isomorphic to $\mathbb{R}^{n_1\times\ldots\times n_d}$, we can simplify the notation by only regarding these spaces of multidimensional arrays, which are introduced further in the next section. Even for finite dimensions, a major obstacle in working with higher order tensors is already apparent: The dimension scales exponentially with respect to the order of the tensor space, i.e. $\mathcal{O}(n^d)$ with $n := \max(n_\mu)$. This exponential scaling is often referred to as the *curse of dimensionality* and circumventing this challenge will be the main subject of Chapter 3.

2.2 The Tensor Spaces $\mathbb{R}^{n_1\times\ldots\times n_d}$

This section is concerned with the spaces $\mathbb{R}^{n_1\times\ldots\times n_d}$ of multidimensional arrays, which will be used throughout this thesis as *the* representative tensor spaces. Its main purpose is to introduce the notation for subsequent chapters and recapitulate some basic properties. In

[1]Note that the order of such tensor spaces depends on the context, as for example $U_1 \otimes (U_2 \otimes U_3) = U_1 \otimes U_2 \otimes U_3$ can be seen as an order two or order three tensor space. In practice this will be no problem, as the context is usually clear.

the rest of this thesis, the term tensor always refers to an element of one of the spaces $\mathbb{R}^{n_1\times\ldots\times n_d}$, unless explicitly stated otherwise.

Intuitively, tensors of $\mathbb{R}^{n_1\times\ldots\times n_d}$ can be considered as the multidimensional generalization of vectors from $\mathbb{R}^n$ and matrices from $\mathbb{R}^{m\times n}$. While vectors and matrices can be interpreted as one- and two-dimensional arrays, respectively, tensors can be interpreted as d-dimensional arrays. More precisely, using the notation $\mathbb{N}_k := \{1, \ldots, k\}$, a vector $\boldsymbol{x} \in \mathbb{R}^n$ can be seen as a mapping

$$\begin{aligned} \boldsymbol{x} : \mathbb{N}_n &\to \mathbb{R} \\ i &\mapsto \boldsymbol{x}[i] \ , \end{aligned}$$

where $\boldsymbol{u}[i]$ denotes the i-th entry of the vector. In the same way, a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ can be seen as a mapping

$$\begin{aligned} \boldsymbol{X} : \mathbb{N}_m \times \mathbb{N}_n &\to \mathbb{R} \\ (i, j) &\mapsto \boldsymbol{X}[i, j] \ . \end{aligned}$$

For fixed order d, this motivates the following definition.

Definition 2.7 (Tensor)**.** A tensor $\mathcal{X} \in \mathbb{R}^{n_1\times\ldots\times n_d}$ of order $d \in \mathbb{N}$ is a mapping

$$\begin{aligned} \mathcal{X} : \mathbb{N}_{n_1} \times \ldots \times \mathbb{N}_{n_d} &\to \mathbb{R} \\ (i_1, \ldots, i_d) &\mapsto \mathcal{X}[i_1, \ldots, i_d] \ . \end{aligned}$$

The number n_μ is called the μ-th dimension of $\mathcal{X}$, $\mathbb{N}_{n_1} \times \ldots \times \mathbb{N}_{n_d}$ its index set and $d \in \mathbb{N}$ is called the *order* of the tensor. In Proposition 2.9, we will show that this definition of the order is consistent with the one for general tensors introduced in the previous section.

From this definition, it is clear that tensors are a generalization of vectors and matrices, as these are order one and order two tensors, respectively. In distinction to these, tensors with order larger than two are referred to as *higher order tensors.* As in the case of vectors and matrices, the entries of a tensor are written as $\mathcal{X}[i_1, \ldots, i_d]$. As a naming convention, we say that the individual indices i_μ correspond to the *μ-th dimension* or *μ-th mode.* The synonyms *μ-th direction*, *μ-th position*, *μ-th axis* and *μ-th site* are common in the literature, but will not be used in this thesis. Note that, as with vectors and matrices, tensors are completely defined by their entries. This allows the entrywise definition of tensors often used within this thesis. For example, let $\mathcal{Y} \in \mathbb{R}^{m\times n}$ be a tensor of order two, then

$$\mathcal{X}[j, i] = \mathcal{Y}[i, j] \quad \forall i, j \ ,$$

defines $\mathcal{X} \in \mathbb{R}^{n\times m}$ to be the transpose of $\mathcal{Y}$ in the matrix sense. As a notational convention, the $\forall i, j$ will usually be omitted in such entrywise definitions, e.g. in this case we only write

$$\mathcal{X}[j, i] = \mathcal{Y}[i, j] \ .$$

As formalized in the following proposition, the set of all tensors with the same function signature, i.e. the same index set, admits a vector space structure using the canonical entrywise addition and scalar multiplication.

Proposition 2.8 (Vector Space $\mathbb{R}^{n_1\times\ldots\times n_d}$)**.** *For every fixed index set $\mathbb{N}_{n_1}\times\ldots\times\mathbb{N}_{n_d}$, let*

$$\mathbb{R}^{n_1\times\ldots\times n_d} := \{\mathcal{X} : \mathbb{N}_{n_1}\times\ldots\times\mathbb{N}_{n_d} \to \mathbb{R}\}$$

be the set of all mappings $\mathbb{N}_{n_1}\times\ldots\times\mathbb{N}_{n_d}\to\mathbb{R}$. Define an addition "+" on $\mathbb{R}^{n_1\times\ldots\times n_d}$ as

$$(\mathcal{X}+\mathcal{Y})[i_1,\ldots,i_d] = \mathcal{X}[i_1,\ldots,i_d] + \mathcal{Y}[i_1,\ldots,i_d] \ ,$$

for all $\mathcal{X},\mathcal{Y}\in\mathbb{R}^{n_1\times\ldots\times n_d}$. The tensor $0\in\mathbb{R}^{n_1\times\ldots\times n_d}$, with $0[i_1,\ldots,i_d]=0\ \forall\ i_1,\ldots,i_d$ is the neutral element of this addition. Define a scalar multiplication "$\cdot$", such that for every scalar $\alpha\in\mathbb{R}$ and $\mathcal{X}\in\mathbb{R}^{n_1\times\ldots\times n_d}$

$$(\alpha\cdot\mathcal{X})[i_1,\ldots,i_d] = \alpha\cdot(\mathcal{X}[i_1,\ldots,i_d]) \ .$$

As is easily verified, $(\mathbb{R}^{n_1\times\ldots\times n_d},+,\cdot)$ forms a real vector space.

The tensor structure of the spaces $\mathbb{R}^{n_1\times\ldots\times n_d}$ is due to the fact that they can be constructed from the real coordinate spaces $\mathbb{R}^{n_\mu}$.

Proposition 2.9. *Each space $\mathbb{R}^{n_1\times\ldots\times n_d}$ can be expressed as the d-fold tensor product of order one tensor spaces*

$$\mathbb{R}^{n_1\times\ldots\times n_d} = \mathbb{R}^{n_1}\otimes\mathbb{R}^{n_2}\otimes\ldots\otimes\mathbb{R}^{n_d} = \bigotimes_{\mu=1}^{d}\mathbb{R}^{n_\mu} \ .$$

In particular, every tensor $\mathcal{X}\in\mathbb{R}^{n_1\times\ldots\times n_d}$ can be expressed as a linear combination of elementary tensors, *i.e.*

$$\mathcal{X} = \sum_j \boldsymbol{x}_{1,j}\otimes\boldsymbol{x}_{2,j}\otimes\ldots\otimes\boldsymbol{x}_{d-1,j}\otimes\boldsymbol{x}_{d,j} \qquad\qquad \boldsymbol{x}_{\mu,j}\in\mathbb{R}^{n_\mu} \ .$$

For the real coordinate spaces, the tensor product can be written as an outer product, i.e. given $\boldsymbol{x}_1\in\mathbb{R}^{n_1},\ldots,\boldsymbol{x}_d\in\mathbb{R}^{n_d}$, we have

$$(\boldsymbol{x}_1\otimes\ldots\otimes\boldsymbol{x}_d)[i_1,\ldots,i_d] = \boldsymbol{x}_1[i_1]\,\boldsymbol{x}_2[i_2]\ldots\boldsymbol{x}_d[i_d] \ .$$

Furthermore, for general tensors $\mathcal{X}\in\mathbb{R}^{n_1\times\ldots\times n_d}$ and $\mathcal{Y}\in\mathbb{R}^{m_1\times\ldots\times m_e}$, we have

$$(\mathcal{X}\otimes\mathcal{Y})[i_1,\ldots,i_d,j_1,\ldots,j_e] = \mathcal{X}[i_1,\ldots,i_d]\cdot\mathcal{Y}[j_1,\ldots,j_e] \ .$$

All results from Section 2.1 naturally extend to the spaces $\mathbb{R}^{n_1\times\ldots\times n_d}$ as well. The induced scalar product and norm are exactly the Frobenius scalar product and norm known from matrices.

Proposition 2.10 (Frobenius Scalar Product and Norm)**.** *Let the real coordinate spaces* $\mathbb{R}^{n_\mu}$ *be equipped with the canonical scalar product. Then, the unique induced scalar product on* $\mathbb{R}^{n_1 \times \ldots \times n_d}$*, i.e. the scalar product for which*

$$\left\langle \bigotimes_{\mu=1}^{d} \boldsymbol{x}_\mu,\ \bigotimes_{\mu=1}^{d} \boldsymbol{y}_\mu \right\rangle = \prod_{\mu=1}^{d} \langle \boldsymbol{x}_\mu,\ \boldsymbol{y}_\mu \rangle$$

holds for all elementary tensors with $\boldsymbol{x}_\mu, \boldsymbol{y}_\mu \in \mathbb{R}^{n_\mu}$*, is the* Frobenius scalar product *defined as*

$$\begin{aligned} \langle \cdot, \cdot \rangle_F : \mathbb{R}^{n_1 \times \ldots \times n_d} \times \mathbb{R}^{n_1 \times \ldots \times n_d} &\to \mathbb{R} \\ (\mathcal{X}, \mathcal{Y}) &\mapsto \sum_{i_1, \ldots, i_d} \mathcal{X}[i_1, \ldots i_d]\, \mathcal{Y}[i_1, \ldots, i_d] \end{aligned}$$

The induced Frobenius norm *is*

$$\|\mathcal{X}\|_F := \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle} = \sqrt{\sum_{i_1, \ldots, i_d} \mathcal{X}[i_1, \ldots, i_d]^2}\ ,$$

which is a crossnorm, meaning that

$$\|\boldsymbol{x}_1 \otimes \ldots \otimes \boldsymbol{x}_d\|_F = \prod_{\mu=1}^{d} \|\boldsymbol{x}_\mu\|$$

holds for all elementary tensors with $\boldsymbol{x}_\mu \in \mathbb{R}^{n_\mu}$*.*

2.2.1 Reshapings, Vectorizations and Matricizations

It is sometimes necessary to reorder the positions of a tensor's modes. Such a so-called reshaping is the high-dimensional analogue of matrix transposition, where the first and second modes are interchanged. However, for higher order tensors, a much richer set of reshapings is possible.

Definition 2.11 (Reshape)**.** For a given order d and a permutation $(\mu_1, \ldots, \mu_d)$ of $(1, \ldots, d)$, a reshape is an isomorphism

$$\operatorname{reshape}_{(\mu_1, \ldots, \mu_d)} : \mathbb{R}^{n_1 \times \ldots \times n_d} \to \mathbb{R}^{n_{\mu_1} \times \ldots \times n_{\mu_d}}$$

defined entrywise as

$$\operatorname{reshape}_{(\mu_1, \ldots, \mu_d)}(\mathcal{X})\,[i_{\mu_1}, \ldots, i_{\mu_d}] := \mathcal{X}[i_1, \ldots, i_d]\ .$$

For order two tensors (matrices), $\operatorname{reshape}_{(2,1)}$ is the matrix transposition. Besides reordering the modes, it is often expedient to combine certain subsets of modes into single modes and thereby interpret a tensor as a lower order object, usually a vector or a matrix. In the following, these operations are formalized as vectorizations and matricizations of a tensor.

Definition 2.12 (Vectorization)**.** Given a bijection $\varphi : \mathbb{N}_{n_1} \times \ldots \times \mathbb{N}_{n_d} \to \mathbb{N}_{m_{\text{vec}}}$, with $m_{\text{vec}} := n_1 n_2 \ldots n_d$, the mapping

$$\begin{aligned} \operatorname{Vec} : \mathbb{R}^{n_1 \times \ldots \times n_d} &\to \mathbb{R}^{m_{\text{vec}}} \\ \operatorname{Vec}(\mathcal{X})[i] &:= \mathcal{X}[\varphi^{-1}(i)] \end{aligned}$$

is called a vectorization. The inverse operation Vec^{-1} is called de-vectorization or tensorization. The choice of the bijective map φ does not matter, provided that the same rule is chosen in all vectorizations (and matricizations). As a convention, in this thesis, a lexicographical ordering of the indices is used, i.e.

$$\begin{aligned} \varphi : \mathbb{N}_{n_1} \times \ldots \times \mathbb{N}_{n_d} &\to \mathbb{N}_{m_{\text{vec}}} \\ (i_1, \ldots, i_d) &\mapsto 1 + \sum_{\mu=1}^{d} (i_\mu - 1) \prod_{\nu<\mu} n_\nu \,. \end{aligned}$$

Definition 2.13 (Matricization)**.** Given a tensor space $\mathbb{R}^{n_1 \times \ldots \times n_d}$, let $\Lambda \subseteq \{1, \ldots, d\}$ denote a subset of the modes and let $\Lambda^C = \{1, \ldots, d\} \setminus \Lambda$ be its complement. Define $m_1 := \prod_{\mu \in \Lambda} n_\mu$ and $m_2 := \prod_{\nu \in \beta} n_\nu$. Given a bijection

$$\begin{aligned} \phi : \mathbb{N}_{n_1} \times \ldots \times \mathbb{N}_{n_d} &\to \mathbb{N}_{m_1} \times \mathbb{N}_{m_2} \\ (i_1, \ldots, i_d) &\mapsto \Big(\varphi_1(i_\mu \mid \mu \in \Lambda),\ \varphi_2(i_\nu \mid \nu \in \Lambda^C) \Big), \end{aligned}$$

the mapping

$$\begin{aligned} \operatorname{Mat}_\Lambda : \mathbb{R}^{n_1 \times \ldots \times n_d} &\to \mathbb{R}^{n_\alpha \times n_\beta} \\ \operatorname{Mat}_\Lambda(\mathcal{X})[i,j] &:= \mathcal{X}[\phi^{-1}(i,j)] \end{aligned}$$

is called the Λ-matricization. The inverse $\operatorname{Mat}_\Lambda^{-1}$ is called the dematricization or tensorization. As with the vectorization, the choice of the bijection ϕ does not matter, provided that the same rule is chosen in every instance. In this thesis, the same lexicographical convention as for the vectorization is used, that is,

$$\begin{aligned} \phi : \mathbb{N}_{n_1} \times \ldots \times \mathbb{N}_{n_d} &\to \mathbb{N}_{m_1} \times \mathbb{N}_{m_2} \\ (i_1, \ldots, i_d) &\mapsto \left(1 + \sum_{\mu \in \Lambda} (i_\mu - 1) \prod_{\substack{\nu<\mu \\ \nu \in \Lambda}} n_\nu,\ 1 + \sum_{\mu \in \Lambda^C} (i_\mu - 1) \prod_{\substack{\nu<\mu \\ \nu \in \Lambda^C}} n_\nu \right) \end{aligned}$$

Remark. It is possible to define tensorizations for any kind of matrix or vector, even if it is not the result of a prior matricization or vectorization. However, this requires to give the dimensions of the resulting tensor and the details of the mapping alongside with the operator, as these are in general not apparent from the context. Instead, in this thesis, the tensorization is always a dematricization, in the sense that it is only applied to matrices where at least one mode of the matrix or vector encodes a tensor structure through a former matricization or vectorization. For all other matrices or modes, the dematricization is simply defined to be the identity.

The two most important matricizations used in this thesis are the μ-mode matricizations $\mathrm{Mat}_{(\mu)}$, where all but the μ-th mode are combined and the so-called μ-mode unfoldings $\mathrm{Mat}_{(1,\dots,\mu)}$, where the first μ modes are combined. For these special matricizations, the following more compact notation is used.

Notation. The μ-mode matricization $\mathrm{Mat}_{(\mu)}$ of a tensor $\mathcal{X}$ is denoted by the same letter in matrix notation (upper case bold), superscripted with (μ), i.e.

$$\boldsymbol{X}^{(\mu)} := \mathrm{Mat}_{(\mu)}\left(\mathcal{X}\right) \ .$$

Notation. The μ-mode unfolding $\mathrm{Mat}_{(1,\dots,\mu)}$ of a tensor $\mathcal{X}$ is denoted by the same letter in matrix notation (upper case bold), superscripted with $\langle\mu\rangle$, i.e.

$$\boldsymbol{X}^{\langle\mu\rangle} := \mathrm{Mat}_{(1,\dots,\mu)}\left(\mathcal{X}\right) \ .$$

Vectorizations and matricizations, as isomorphisms, are uniquely invertible. As long as the dimensions are evident from context, this allows their use also on the left side of assignments. For example, given tensors $\mathcal{Z} \in \mathbb{R}^{m\times n\times m\times n}$ and $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{m\times n\times p}$, then

$$\boldsymbol{Z}^{\langle 2\rangle} = \boldsymbol{X}^{\langle 2\rangle}\left(\boldsymbol{Y}^{\langle 2\rangle}\right)^T$$

is equivalent to

$$\mathcal{Z}[i_1, i_2, i_3, i_4] = \sum_j \mathcal{X}[i_1, i_2, j]\ \mathcal{Y}[i_3, i_4, j] \ .$$

2.2.2 Contractions

Apart from addition and scalar multiplication, there are several further operations that can be generalized from matrices to higher order tensors. Amongst the most important are the matrix-vector and matrix-matrix multiplications, whose multidimensional counterpart are the tensor *contractions*.

Definition 2.14 (Tensor Contraction)**.** Let $\mathcal{X} \in \mathbb{R}^{n_1\times\dots\times n_d}$ and $\mathcal{Y} \in \mathbb{R}^{m_1\times\dots\times m_e}$ be two tensors of order d and e, respectively. The contraction $\mathcal{X} *_{\mu,\nu} \mathcal{Y}$ of the μ-th mode of $\mathcal{X}$ with the ν-th mode of $\mathcal{Y}$ is defined entrywise as

$$\begin{aligned}&(\mathcal{X} *_{\mu,\nu} \mathcal{Y})[i_1, \dots, i_{\mu-1}, i_{\mu+1}, \dots, i_d, j_1, \dots, j_{\nu-1}, j_{\nu+1}, \dots, j_e]\\ &:= \sum_{p=1}^{n_\mu} \mathcal{X}[i_1, \dots, i_{\mu-1}, p, i_{\mu+1}, \dots, i_d]\ \mathcal{Y}[j_1, \dots, j_{\nu-1}, p, j_{\nu+1}, \dots, j_e] \ .\end{aligned}$$

This definition can equivalently be expressed via matricizations as

$$\mathcal{X} *_{\mu,\nu} \mathcal{Y} = \mathrm{Mat}^{-1}\left(\mathrm{Mat}_{(\mu)}(\mathcal{X})^T \mathrm{Mat}_{(\nu)}(\mathcal{Y})\right) = \mathrm{Mat}^{-1}\left((\boldsymbol{X}^{(\mu)})^T \boldsymbol{Y}^{(\nu)}\right) \ .$$

The resulting tensor $(\mathcal{X} *_{\mu,\nu} \mathcal{Y}) \in \mathbb{R}^{n_1\times\dots\times n_{\mu-1}\times n_{\mu+1}\times\dots\times n_d\times m_1\times\dots m_{\nu-1}\times m_{\nu+1}\times\dots\times m_e}$ is of order $d+e-2$. Note that in order for this operation to be well-defined, $n_\mu = m_\nu$ must hold, i.e. the μ-th index set of $\mathcal{X}$ and the ν-th index set of $\mathcal{Y}$ must coincide.

The contraction of multiple mode pairs is defined analogously, with the sum being performed over all indices belonging to the corresponding pairs. That is, for $\mathcal{X}$ and $\mathcal{Y}$ as above, the contraction $\mathcal{X} *_{(\mu_1,\ldots,\mu_s),(\nu_1,\ldots,\nu_s)} \mathcal{Y}$ of the mode pairs $\mu_1/\nu_1, \mu_2/\nu_2, \ldots, \mu_s/\nu_s$ is defined as

$$\begin{aligned}&\Big(\mathcal{X} *_{(\mu_1,..,\mu_s),(\nu_1,..,\nu_s)} \mathcal{Y}\Big)\,[i_1,..,i_{\mu_1-1},i_{\mu_1+1},..,i_{\mu_s-1},i_{\mu_s+1},..,i_d,j_1,..,j_{\nu_1-1},j_{\nu_1+1},..,j_e]\\ &= \sum_{k_1,\ldots,k_s} \mathcal{X}[i_1,\ldots,k_1,i_{\mu_1+1},\ldots,k_s,i_{\mu_s+1},\ldots,i_d]\; \mathcal{Y}[j_1,\ldots,k_1,j_{\nu_1+1},\ldots,k_s,j_{\mu_s+1},\ldots,j_e]\,.\end{aligned}$$

Here, it is again assumed that $n_{\mu_1} = m_{\nu_1}, \ldots, n_{\mu_s} = m_{\nu_s}$ holds.

Finally, it is convenient to define the contraction of an empty set of mode pairs $\mathcal{X} * \mathcal{Y}$, equivalent to the dyadic product, as

$$(\mathcal{X} * \mathcal{Y})\,[i_1,\ldots,i_d,j_1,\ldots,j_e] = \mathcal{X}[i_1,\ldots,i_d]\;\mathcal{Y}[j_1,\ldots,j_e]\ .$$

An important property inherited from the matrix-matrix product is the associativity, which also holds for arbitrary contractions, if the modes involved in the contraction are kept fixed. This is obvious from the entrywise definition of the contractions. For example, given $\mathcal{X} \in \mathbb{R}^{n_1\times n_2\times n_3}, \mathcal{Y} \in \mathbb{R}^{m_2\times n_2\times p}, \mathcal{Z} \in \mathbb{R}^{m_1\times m_2\times m_3}$, it is clear that

$$\begin{aligned}\sum_{j=1}^{n_2} \mathcal{X}[i_1,j,i_2]\left(\sum_{k=1}^{m_2} \mathcal{Y}[k,j,i_3]\mathcal{Z}[i_4,k,i_5]\right) &= \sum_{j=1}^{n_2}\sum_{k=1}^{m_2} \mathcal{X}[i_1,j,i_2]\mathcal{Y}[k,j,i_3]\mathcal{Z}[i_4,k,i_5]\\ &= \sum_{k=1}^{m_2}\left(\sum_{j=1}^{n_2} \mathcal{X}[i_1,j,i_2]\mathcal{Y}[k,j,i_3]\right)\mathcal{Z}[i_4,k,i_5]\end{aligned}$$

holds. However, there are unfortunately some issues with the notation, in particular, the "$*$" notation is *not* associative, because the positions of the modes within the involved tensors change. Note that in the above setting,

$$\mathcal{X} *_{2,1} \underbrace{(\mathcal{Y} *_{1,2} \mathcal{Z})}_{\in\mathbb{R}^{n_2\times p\times m_1\times m_3}} \tag{2.1}$$

is a well defined tensor from $\mathbb{R}^{n_1\times n_3\times p\times m_1\times m_3}$, while

$$(\mathcal{X} *_{2,1} \mathcal{Y}) *_{1,2} \mathcal{Z} \tag{2.2}$$

is not even well-defined, because the dimensions do not match. This is due to the fact that in (2.1), the left contraction acts on the second mode of $\mathcal{X}$ and the *second* mode of the original $\mathcal{Y}$. In the tensor $(\mathcal{Y} *_{1,2} \mathcal{Z})$, this second mode of $\mathcal{Y}$ is moved to the first position, which is why the left contraction has to act on the first mode of this resulting tensor. In (2.2), the left contraction acts on the *first* mode of $\mathcal{Y}$ instead of the second, which, due to the different dimensions, is ill-defined in general. To circumvent this issue, the syntax has to be adjusted according to the position of the actual modes the contractions act upon, i.e.

$$\mathcal{X} *_{2,1} (\mathcal{Y} *_{1,2} \mathcal{Z}) = (\mathcal{X} *_{2,2} \mathcal{Y}) *_{3,2} \mathcal{Z}\ .$$

There are two special kinds of contractions which are predominantly used in this thesis and for which explicit notations are introduced. The first is the μ-th mode product, which is a contraction between a tensor and a matrix followed by a reshape which restores the tensor's mode order.

Definition 2.15 (μ-th Mode Product)**.** Given a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ and a matrix $\boldsymbol{A} \in \mathbb{R}^{n_\mu \times m}$, the μ-th mode product between $\mathcal{X}$ and $\boldsymbol{A}$ is defined as

$$\begin{aligned} \times_\mu : \mathbb{R}^{n_1 \times \ldots \times n_d} \times \mathbb{R}^{n_\mu \times m} &\to \mathbb{R}^{n_1 \times \ldots \times n_{\mu-1} \times m \times n_{\mu+1} \times \ldots \times n_d} \\ \left(\mathcal{X} \times_\mu \boldsymbol{A}\right)[i_1, \ldots, i_d] &:= \sum_{j=1}^{n_\mu} \mathcal{X}[i_1, \ldots, i_{\mu-1}, j, i_{\mu+1}, \ldots, i_d] \boldsymbol{A}[j, i_\mu] \ . \end{aligned}$$

An equivalent definition is given by

$$\mathcal{X} \times_\mu \boldsymbol{A} := \operatorname{reshape}_{(1,\ldots,\mu-1,d,\mu,\ldots,d-1)} \left(\mathcal{X} *_{\mu,1} \boldsymbol{A}\right) \ .$$

One easily verifies that mode products on different modes commute, i.e. for $\mu \neq \nu$

$$\mathcal{X} \times_\mu \boldsymbol{A} \times_\nu \boldsymbol{B} = \mathcal{X} \times_\nu \boldsymbol{B} \times_\mu \boldsymbol{A}$$

holds for any $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$, $\boldsymbol{A} \in \mathbb{R}^{n_\mu \times m_A}$ and $\boldsymbol{B} \in \mathbb{R}^{n_\nu \times m_B}$. Successive mode products on the same mode can instead be combined to a matrix-matrix product, i.e.

$$\mathcal{X} \times_\mu \boldsymbol{A} \times_\mu \boldsymbol{B} = \mathcal{X} \times_\mu (\boldsymbol{A}\boldsymbol{B})$$

holds for any $1 \leq \mu \leq d, \mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}, \boldsymbol{A} \in \mathbb{R}^{n_\mu \times m_A}$ and $\boldsymbol{B} \in \mathbb{R}^{m_A \times m_B}$.

The second special kind of contractions are those, in which the last k modes of the left operand are contracted with the first k modes of the right operand, as formalized in the following.

Notation. Given tensors $\mathcal{X} \in \mathbb{R}^{n_1 \ldots, n_{d-k} \times m_1 \times \ldots \times m_k}$ and $\mathcal{Y} \in \mathbb{R}^{m_1 \times \ldots \times m_k \times p_1 \ldots, p_{e-k}}$ of orders d and e, respectively, the contraction between the last k modes of $\mathcal{X}$ and the first k modes of $\mathcal{Y}$ is written as

$$\mathcal{X} \circ_k \mathcal{Y} := \mathcal{X} *_{(d+1,\ldots,d+k),(1,\ldots,k)} \mathcal{Y} = \operatorname{Mat}^{-1}\left(\left(\boldsymbol{X}^{\langle k \rangle}\right)^T \boldsymbol{Y}^{\langle k \rangle}\right) \ .$$

If only one mode pair is contracted, the index is usually dropped, i.e. $\circ_1$ becomes $\circ$.

For order one and two tensors, i.e. vectors and matrices, this last notation is exactly the matrix-vector and matrix-matrix product, where the circle is usually also omitted. In particular, given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}$,

$$\boldsymbol{x} *_{1,1} \boldsymbol{y} = \boldsymbol{x} \circ \boldsymbol{y} = \boldsymbol{x}^T \boldsymbol{y}$$

reproduces the vector scalar product,

$$\boldsymbol{A} *_{2,1} \boldsymbol{x} = \boldsymbol{A} \circ \boldsymbol{x} = \boldsymbol{A}\boldsymbol{x}$$

reproduces the matrix-vector product and

$$\boldsymbol{A} *_{2,1} \boldsymbol{B} = \boldsymbol{A} \circ \boldsymbol{B} = \boldsymbol{AB}$$

reproduces the matrix-matrix product. Similarly to the vector case, the Frobenius scalar product and norm from Proposition 2.10 can be expressed as full contractions, i.e.

$$\langle \mathcal{X}, \mathcal{Y} \rangle_F = \mathcal{X} *_{(1,\dots,d),(1,\dots,d)} \mathcal{Y} = \mathcal{X} \circ_d \mathcal{Y} \ ,$$
$$\|\mathcal{X}\|_F = \sqrt{\mathcal{X} *_{(1,\dots,d),(1,\dots,d)} \mathcal{X}} = \sqrt{\mathcal{X} \circ_d \mathcal{X}} \ ,$$

holds for all $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{n_1 \times \dots \times n_d}$. Note that the $\circ$ notation is associative if the order of each tensor is larger or equal to the sum of modes contracted from both sites, i.e.

$$(\mathcal{X} \circ_p \mathcal{Y}) \circ_q \mathcal{Z} = \mathcal{X} \circ_p (\mathcal{Y} \circ_q \mathcal{Z})$$

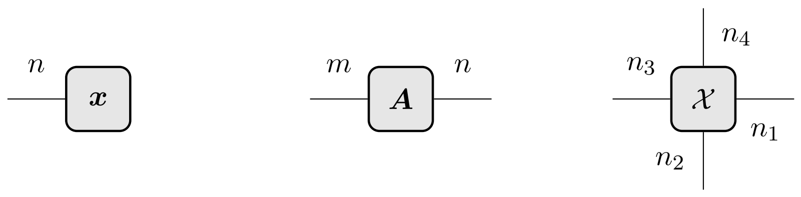
holds if the order of $\mathcal{Y}$ is at least $p + q$.

2.3 Tensor Networks and Diagrammatic Notation

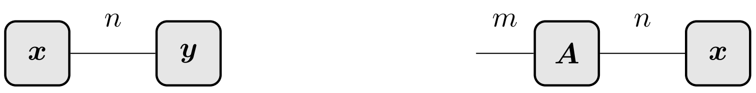
In this section, we introduce the notion of a tensor network, which is an important formal construct for several tensor decompositions discussed in Chapter 3.

Definition 2.16 (Tensor Network)**.** A tensor network is defined by a (finite) set of tensors T and a set of contractions C acting on (unique) pairs of these tensors. The structure of a tensor network can be visualized by a graph, with nodes corresponding to T and edges corresponding to C, i.e. two nodes are connected if and only if there is a contraction performed between them. The tensor that is obtained by performing all contractions C is said to be the tensor represented by the tensor network. In case the corresponding graph is not connected, dyadic contractions, i.e. contractions with an empty set of contracted nodes, are performed between all connected components. As shown in the previous section, the order of the contractions does not matter.

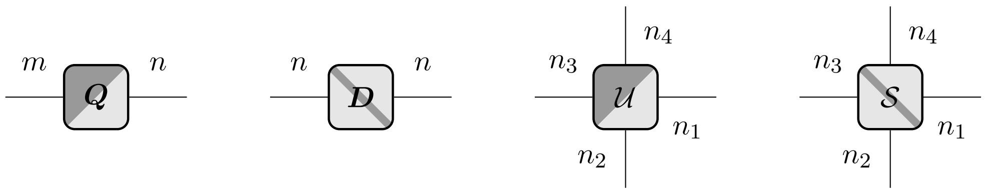
The concept of tensor networks is especially helpful, because it allows to visualize larger tensor expressions using the graph structure. In this diagrammatic notation, a tensor is depicted as a box with edges corresponding to each of its modes. If appropriate, the dimension of the corresponding mode is given as well. From left to right, the following shows this for an order one tensor (vector) $\boldsymbol{x} \in \mathbb{R}^n$, an order two tensor (matrix) $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and an order four tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$.



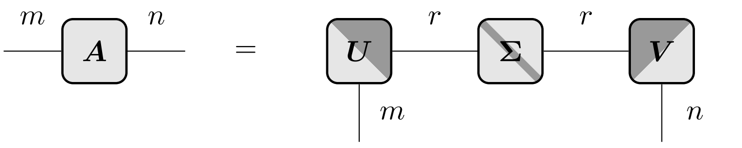
If a contraction is performed between two modes of two tensors, the corresponding edges are joined. The following exemplifies this for the inner product of two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and a matrix-vector product of $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with $\boldsymbol{x} \in \mathbb{R}^n$.



Unfortunately, the visualization as a graph loses the information on which modes the contractions act. It is therefore only used in the following if the modes are evident from context. There are two special cases concerning orthogonal and diagonal matrices. If a specific matricization of a tensor yields an orthogonal or diagonal matrix, the tensor is depicted by a half filled box, or a box with a diagonal bar, respectively. The half filling and the diagonal bar both divide the box in two halves. The edges joined to either half correspond to the mode sets of the matricization which yields the orthogonal or diagonal matrix.[2] From left to right, the following shows this for a right-orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{m \times n}$, a diagonal matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$, a tensor $\mathcal{U} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$, such that the matricization $\boldsymbol{U}^{\langle 2 \rangle} = \mathrm{Mat}_{(1,2)}(\mathcal{U}) \in \mathbb{R}^{n_1 n_2 \times n_3 n_4}$ is left-orthogonal and a tensor $\mathcal{S} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$, such that the matricization $\mathrm{Mat}_{(1,4)}(\mathcal{S}) \in \mathbb{R}^{n_1 n_4 \times n_2 n_3}$ is diagonal.



As an example, the diagrammatic notation can be used to depict the singular value decomposition $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with rank r, as shown in the following.



More complex networks will appear in the following chapter as a means to represent higher order tensors.

[2]For left- or right-orthogonal matrices (see Definition 3.1), the bright side indicates the larger dimension.

3 Tensor Decompositions

The main obstacle in working with higher order tensors is the exponential scaling of the dimension with respect to the order, the so-called *curse of dimensionality*. Even for moderately sized local dimensions, this causes the computational costs to rapidly escalate. A prominent example are many particle quantum states, for which the order of the tensor Hilbert space increases with the number of particles. Even taking the very simple model of a spin chain, for which the local dimension $n = 2$ is minimal, the tensor Hilbert space for 100 particles has order $d = 100$ and its dimension is $2^{100} \approx 10^{30}$. Handling this space exceeds the capabilities of current datacenters by far, as even storing *one* state in single precision would require more than 10^{12} Exabytes of memory, which is by orders of magnitude greater than the storage capacity of around 295 Exabytes available to humankind as a whole (in 2007, see [1]). Therefore, in order to work with such high order tensors, some kind of data sparse representation is needed. This format should allow a rich set of common operations to be computed while staying in this data sparse representation. Very successful approaches to this are *low rank tensor decompositions*, which are introduced in this chapter. While there are several quite different tensor decomposition formats, the fundamental idea of all these formats is to generalize the singular value decomposition and the notion of rank from matrices to higher order tensors.

This chapter starts in Section 3.1 with a brief recapitulation of the matrix rank, low rank matrix decompositions and possible operations which preserve a low rank matrix structure. This establishes the main ideas and motivates similar approaches for the multidimensional generalizations in the subsequent sections. In Sections 3.2, 3.3 and 3.4, three of the most important tensor decompositions are introduced in the chronological order of their invention. The focus will be on the *Tensor Train (TT)* decomposition, as it is used the most in the remainder of the thesis. Section 3.5 provides a brief overview of even more general decomposition formats. Finally, Section 3.6 gives a summary and comparison between the different formats.

As a prerequisite, let us begin with the somewhat non-standard definition of left- and right-orthogonal matrices and tensors that will be important throughout this thesis.

Definition 3.1 (Left-/Right-orthogonal Matrix)**.** A matrix $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ is orthogonal iff its rows and columns are orthogonal unit vectors, that is

$$\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}_n \ .$$

A matrix $\boldsymbol{U} \in \mathbb{R}^{m \times n}$ is left-/right-orthogonal iff its columns/rows are orthogonal unit vectors. That is, $\boldsymbol{U}$ is left-orthogonal iff $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_n$ and right-orthogonal iff $\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}_m$. Note that every orthogonal matrix is both left- and right-orthogonal and every left-/right-orthogonal square matrix is also (right-/left-)orthogonal.

Definition 3.2 (Left-/Right-orthogonal Tensor)**.** A tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ is left-orthogonal, iff the $d-1$-mode unfolding $\boldsymbol{X}^{<d-1>}$ yields a left-orthogonal matrix. Analogously, $\mathcal{X}$ is right-orthogonal, iff the 1-mode unfolding $\boldsymbol{X}^{<1>}$ yields a right-orthogonal matrix.

3.1 Low Rank Matrices

In this section, a brief summary is given for the most important definitions and results concerning the matrix rank and low rank matrix decompositions. This summary serves the purpose of motivating and explaining the reasoning behind the low rank tensor decompositions of the subsequent sections. For a detailed introduction to numerical linear algebra, the reader is referred to the standard reference by Golub and Van Loan [71], where proofs for all propositions of this section can be found, for which no explicit reference is given.

3.1.1 Matrix Rank and Singular Value Decomposition

There are various equivalent ways to define the rank of a matrix, some of which are given in the following. These different approaches will turn out to be important, as it is later shown that they are not equivalent for tensors and lead to different definitions of the tensor rank.

Definition 3.3 (Matrix Rank)**.** A matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ has rank r iff:

(I) There are exactly r linearly independent columns in $\boldsymbol{A}$.

(II) There are exactly r linearly independent rows in $\boldsymbol{A}$.

(III) The image of the linear map induced by $\boldsymbol{A}$ has dimension r.

(IV) The quotient of $\mathbb{R}^m$ by the kernel of $\boldsymbol{A}$ has dimension r, i.e. $\dim(\mathbb{R}^m / \ker(\boldsymbol{A})) = r$.

(V) r is the smallest number, such that there exist vectors $\boldsymbol{u}_i \in \mathbb{R}^m$, $\boldsymbol{v}_i \in \mathbb{R}^n$ and real numbers $\sigma_i > 0$, for $i = 1, \dots, r$, such that

$$\boldsymbol{A} = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T \ .$$

(VI) r is the smallest number, such that there exist r-dimensional subspaces $V \subseteq \mathbb{R}^m$ and $U \subseteq \mathbb{R}^n$, such that $\boldsymbol{A}$ is an element of the induced tensor space $V \otimes U \subseteq \mathbb{R}^{m \times n}$.

(VII) r is the smallest number, such that there exist left-orthogonal matrices $\boldsymbol{U} \in \mathbb{R}^{m\times r}$, $\boldsymbol{V} \in \mathbb{R}^{n\times r}$ and a diagonal matrix $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \sigma_2, \dots, \sigma_r) \in \mathbb{R}^{r\times r}$, with

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \ .$$

The notion of matrix rank is closely linked to the *singular value decomposition* (SVD), which in its general form is defined as follows.

Theorem 3.4 (Singular Value Decomposition (SVD))**.** *Every matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ can be factorized by a singular value decomposition*

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

where $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ are orthogonal matrices and

$$\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \sigma_2, \dots, \sigma_{\min(m,n)}) \in \mathbb{R}^{m\times n}$$

is a rectangular diagonal matrix with ordered, non-negative entries $\sigma_1 \geq \sigma_2 \geq \dots \geq \boldsymbol{\sigma}_{\min(m,n)} \geq 0$. These entries are the singular values of $\boldsymbol{A}$. The number of non-zero singular values is equal to the rank of $\boldsymbol{A}$. The Singular Value Decomposition is in general not unique, but the matrix $\boldsymbol{\Sigma}$ and the singular values σ_i are.

As the singular values are unique, we can use the notation $\sigma_i(\boldsymbol{A})$ to denote the i-th singular value of a matrix $\boldsymbol{A}$ without explicitly giving the complete SVD. There is a particularly noteworthy interpretation of the spaces spanned by $\boldsymbol{U}$ and $\boldsymbol{V}$: Let $\hat{\boldsymbol{A}} : \mathbb{R}^n \to \mathbb{R}^m$ be the linear map induced by $\boldsymbol{A}$. Then, the column vectors of $\boldsymbol{V} = (\boldsymbol{v}_1, \dots, \boldsymbol{v}_n)$ form an orthogonal basis of the domain $\mathbb{R}^n$ of $\hat{\boldsymbol{A}}$ and the column vectors of $\boldsymbol{U} = (\boldsymbol{u}_1, \dots, \boldsymbol{u}_m)$ form a basis of its codomain. Furthermore, let r be the rank of $\boldsymbol{A}$, then the vectors $\boldsymbol{v}_{r+1}, \dots \boldsymbol{v}_n$ are a basis of the kernel of $\hat{\boldsymbol{A}}$ and $\boldsymbol{u}_{r+1}, \dots, \boldsymbol{u}_m$ form a basis of $\mathbb{R}^m/\operatorname{im}(\hat{\boldsymbol{A}})$. The corresponding singular values $\sigma_{r+1}, \dots, \sigma_{\min(m,n)}$ are all zero. For an exact representation of $\boldsymbol{A}$, the basis of the kernel and of $\mathbb{R}^m/\operatorname{im}(\hat{\boldsymbol{A}})$ are not needed. Therefore, by truncating the last $m-r$ columns of $\boldsymbol{U}$ and rows of $\boldsymbol{\Sigma}$, as well as the last $n-r$ columns of $\boldsymbol{\Sigma}$ and rows of $\boldsymbol{V}^T$, one acquires an exact decomposition of $\boldsymbol{A}$ of the form

$$\boldsymbol{A} = \tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T \ , \tag{3.1}$$

with left-orthogonal $\tilde{\boldsymbol{U}} \in \mathbb{R}^{m\times r}$, $\tilde{\boldsymbol{V}} \in \mathbb{R}^{n\times r}$ and diagonal $\tilde{\boldsymbol{\Sigma}} = \operatorname{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{r\times r}$. A decomposition of this form is usually referred to as a *compact singular value decomposition* in the literature. This compact SVD is the most useful for our purposes and in the remainder of this thesis we will always assume a compact SVD, unless explicitly stated otherwise.

The Singular Value Decomposition also allows obtaining a rank $r' < r$ approximation $\hat{H}_{r'}(\boldsymbol{A})$ of $\boldsymbol{A}$. To this end, a SVD is calculated and then all but the first (largest) r' singular values are set to zero. As the singular values are in descending order by size, this means that the $\min(m,n) - r'$ smallest singular values are neglected. Analogously to (3.1), one obtains

$$\hat{H}_{r'}(\boldsymbol{A}) = \boldsymbol{U}'\boldsymbol{\Sigma}'\boldsymbol{V}'^T \ , \tag{3.2}$$

where the left-orthogonal matrices $\boldsymbol{U}' \in \mathbb{R}^{m\times r'}$ and $\boldsymbol{V}' \in \mathbb{R}^{n\times r'}$ are obtained by truncating $\boldsymbol{U}$ respectively $\boldsymbol{V}$ to r' columns. $\boldsymbol{\Sigma}'$ is obtained from $\boldsymbol{\Sigma}$ by truncation to r' rows and columns. The non-linear operator $\hat{H}_{r'}$ introduced here is the singular value hard thresholding operator, or simply hard thresholding operator. The corresponding decomposition (3.2) is referred to as the *rank r' truncated SVD* of $\boldsymbol{A}$. The following theorem states that for a fixed rank r', this approximation is in fact the best one possible. It is usually attributed to Eckart and Young [72], although it was already known to E. Schmidt in 1907.[1]

Theorem 3.5 (Eckart–Young)**.** *For every matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, the rank r' approximation obtained by the hard thresholding operator is a best rank r' approximation with respect to both the Frobenius and the spectral norm. That is,*

$$\left\|\boldsymbol{A} - \hat{H}_{r'}(\boldsymbol{A})\right\| = \min\{\|\boldsymbol{A} - \boldsymbol{X}\| \mid \boldsymbol{X} \in \mathbb{R}^{m\times n},\ \operatorname{rank}(\boldsymbol{X}) \le r'\} \ ,$$

where $\|\cdot\|$ is either the Frobenius norm $\|\cdot\|_F$, or the spectral norm $\|\cdot\|_2$. The error is given by

$$\left\|\boldsymbol{A} - \hat{H}_{r'}(\boldsymbol{A})\right\|_F = \left(\sum_{k>r'} \sigma_k(\boldsymbol{A})^2\right)^{1/2} \ ,$$

respectively

$$\left\|\boldsymbol{A} - \hat{H}_{r'}(\boldsymbol{A})\right\|_2 = \sigma_{r'+1}(\boldsymbol{A}) \ .$$

3.1.2 Important Properties

This section summarizes several important properties of the singular values of sub-matrices and the effect of orthogonal transformations.

Lemma 3.6 (Horn and Johnson [74, Corollary 3.1.3])**.** *Given $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, let $\boldsymbol{A}_p$ denote a submatrix of $\boldsymbol{A}$ obtained by deleting a total of p rows and/or columns from $\boldsymbol{A}$. Then, for $1 \le k \le \min(m,n)$,*

$$\sigma_k(\boldsymbol{A}) \ge \sigma_k(\boldsymbol{A}_p) \ge \sigma_{k+p}(\boldsymbol{A})$$

holds, where we set $\sigma_k(\boldsymbol{A}_p) = 0$ if there are fewer than k rows or columns left.

Proposition 3.7. *For every matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and every right-orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times p}$ with $n \le p$ and $\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I}_n$, the first n singular values of $\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{Q}$ coincide and all further singular values of $\boldsymbol{A}\boldsymbol{Q}$ are zero.*

Proof. Let $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ be a SVD of $\boldsymbol{A}$, then $\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{\Sigma}\tilde{\boldsymbol{V}}^T$ is a valid SVD of $\boldsymbol{A}\boldsymbol{Q}$, because $\boldsymbol{Q}^T\boldsymbol{V} = \tilde{\boldsymbol{V}} \in \mathbb{R}^{n\times p}$ is left-orthogonal. Since the unique diagonal matrix $\boldsymbol{\Sigma}$ is unchanged, the (unique) singular values coincide. □

[1]Unfortunately the original paper of E. Schmidt is not available to the author. However, see the survey of Stewart [73] for some interesting historical notes on the singular value decomposition.

Proposition 3.8 (Adapted from [75, Theorem 2.1]). *Given a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and a left-orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times p}$ with $p \leq n$ and $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}_p$, for the singular values of $\boldsymbol{AQ}$, it holds that*

$$\sigma_k(\boldsymbol{AQ}) \leq \sigma_k(\boldsymbol{A})$$

for all $1 \leq k \leq \min(m, p)$.

Proof. Let V be the subspace spanned by the orthonormal columns of $\boldsymbol{Q}$ and let $V^\perp$ be its orthogonal complement. Let $\boldsymbol{W}$ be a matrix whose columns form an orthogonal basis of $V^\perp$. Set $\boldsymbol{M} = [\boldsymbol{Q} \quad \boldsymbol{W}] \in \mathbb{R}^{n\times n}$, which by construction is an orthogonal matrix. Therefore, $\boldsymbol{A}$ and $\boldsymbol{AM} = [\boldsymbol{AQ} \quad \boldsymbol{AW}]$ have the same singular values. Now, $\boldsymbol{AQ}$ can be obtained by deletion of rows from $\boldsymbol{AM}$ and the result directly follows from Proposition 3.6. $\square$

3.1.3 Calculation of a Decomposition

An important practical result is that the SVD of a given matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ can be calculated efficiently, using for example the algorithm described in the reference by Trefethen and Bau III [76]. In particular, for every fixed precision, the computational costs can be bounded by $\mathcal{O}(nm \min(n, m))$, making the SVD an extremely powerful numerical tool. The algorithm is *rank revealing*, i.e. no information about the rank of $\boldsymbol{A}$ is needed in advance.

3.1.4 Low Rank Matrices

The set of $\mathbb{R}^{m\times n}$ matrices with fixed rank r

$$\mathcal{M}_r(\mathbb{R}^{m\times n}) := \{\boldsymbol{X} \in \mathbb{R}^{m\times n} \mid \operatorname{rank}(\boldsymbol{X}) = r\}$$

is not a linear space, because in general the sum of two rank r matrices has a rank between zero and $2r$. Although not a linear space, it can be shown that $\mathcal{M}_r(\mathbb{R}^{m\times n})$ allows a manifold structure. This is done for example by Helmke and Moore [77], who prove the following proposition.

Proposition 3.9 (Fixed Rank Matrix Manifold [77, Chapter 5, Prop. 1.14]). *For every m, $n > 1$, $\mathcal{M}_r(\mathbb{R}^{m\times n})$ is a smooth manifold of dimension $r(m+n-r)$. The tangent space $\mathbb{T}_{\boldsymbol{A}}\mathcal{M}_r(\mathbb{R}^{m\times n})$ at an element $\boldsymbol{A} \in \mathcal{M}_r(\mathbb{R}^{m\times n})$ is given by*

$$\mathbb{T}_{\boldsymbol{A}}\mathcal{M}_r(\mathbb{R}^{m\times n}) \cong \{\boldsymbol{\Delta}_1\boldsymbol{A} + \boldsymbol{A}\boldsymbol{\Delta}_2 \mid \boldsymbol{\Delta}_1 \in \mathbb{R}^{m\times m}, \boldsymbol{\Delta}_2 \in \mathbb{R}^{n\times n}\} \ .$$

Note that $\mathcal{M}_r(\mathbb{R}^{m\times n})$ is not a closed set. As a trivial counterexample, consider an arbitrary rank r matrix $\boldsymbol{A} \in \mathcal{M}_r(\mathbb{R}^{m\times n})$, then $\boldsymbol{A}_k = \frac{1}{k}\boldsymbol{A} \in \mathcal{M}_r(\mathbb{R}^{m\times n})$ is a sequence which converges to $\boldsymbol{0} \notin \mathcal{M}_r(\mathbb{R}^{m\times n})$. The closure of $\mathcal{M}_r(\mathbb{R}^{m\times n})$ is the set

$$\mathcal{M}_{\leq r}(\mathbb{R}^{m\times n}) := \{\boldsymbol{X} \in \mathbb{R}^{m\times n} \mid \operatorname{rank}(\boldsymbol{X}) \leq r\}$$

of matrices with rank at most r. The proof works by characterizing $\mathcal{M}_{\leq r}(\mathbb{R}^{m\times n})$ as those matrices for which all $(r+1)\times(r+1)$ minors vanish. As taking the minor is a continuous mapping, the preimage of $\{0\}$ is closed for every minor and so is $\mathcal{M}_{\leq r}(\mathbb{R}^{m\times n})$, as the intersection of the preimages of all $(r+1)\times(r+1)$ minors. As the rank of any matrix can be increased by arbitrarily small distortions, it is clear that $\mathcal{M}_{r}(\mathbb{R}^{m\times n})$ is dense in $\mathcal{M}_{\leq r}(\mathbb{R}^{m\times n})$.

3.1.5 Computational Aspects

From a numerical point of view, the singular value decomposition allows efficient storage of matrices with small rank. In particular, representing a matrix $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ as its SVD (3.1) only requires to store the $mr+r+rn$ entries of the matrices $\boldsymbol{U}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{V}$, instead of the mn entries of the full matrix. If the rank is small, i.e. if $r\ll\min(m,n)$, this allows a significant reduction in storage requirements. Additionally, there is a rich set of operations which can be computed at significantly reduced cost if the operands are given in a low rank format. In the following, several of these low rank operations are introduced. For the remainder of this section, let $\boldsymbol{A},\bar{\boldsymbol{A}}\in\mathbb{R}^{m\times n}$ be matrices of rank r and $\bar{r}$, respectively. Apart from the normalization paragraph, we assume in the following that both matrices are given in their respective (compact) singular value decomposition, i.e. $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T=\boldsymbol{A}$ and $\bar{\boldsymbol{U}}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{V}}^T=\bar{\boldsymbol{A}}$, with left-orthogonal $\boldsymbol{U},\bar{\boldsymbol{U}},\boldsymbol{V},\bar{\boldsymbol{V}}$ and diagonal $\boldsymbol{\Sigma},\bar{\boldsymbol{\Sigma}}$ with decreasing entries.

Normalization, (Re-)Orthogonalization Assume that $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ is not given in its SVD, but in a low rank decomposition $\boldsymbol{A}=\tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T$ with $\tilde{\boldsymbol{U}}\in\mathbb{R}^{m\times\tilde{r}},\tilde{\boldsymbol{\Sigma}}\in\mathbb{R}^{\tilde{r}\times\tilde{r}},\tilde{\boldsymbol{V}}^T\in\mathbb{R}^{\tilde{r}\times n}$. Here, $\tilde{\boldsymbol{U}}$ and $\tilde{\boldsymbol{V}}$ are not necessarily orthogonal, $\tilde{\boldsymbol{\Sigma}}$ is not necessarily diagonal and $\tilde{r}$ might not be the actual rank of $\boldsymbol{A}$. Transforming this representation to a valid SVD of $\boldsymbol{A}$ is referred to as reorthogonalization. One possible method is to compute the QR-decompositions $\boldsymbol{Q}_L\boldsymbol{R}_L=\tilde{\boldsymbol{U}}$ and $\boldsymbol{Q}_R\boldsymbol{R}_R=\tilde{\boldsymbol{V}}$, calculate $\boldsymbol{X}=\boldsymbol{R}_L\tilde{\boldsymbol{\Sigma}}\boldsymbol{R}_R^T$ and a SVD thereof

$$\boldsymbol{U}_X\boldsymbol{D}\boldsymbol{V}_X^T=\boldsymbol{X}\ .$$

Finally, setting $\boldsymbol{U}:=\boldsymbol{Q}_L\boldsymbol{U}_X$ and $\boldsymbol{V}:=\boldsymbol{Q}_R\boldsymbol{V}_X$ gives

$$\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T=\boldsymbol{Q}_L\boldsymbol{U}_X\boldsymbol{D}\boldsymbol{V}_X^T\boldsymbol{Q}_R^T=\boldsymbol{Q}_L\boldsymbol{X}\boldsymbol{Q}_R^T=\boldsymbol{Q}_L\boldsymbol{R}_L\tilde{\boldsymbol{\Sigma}}\boldsymbol{R}_R^T\boldsymbol{Q}_R^T=\tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T=\boldsymbol{A}\ ,$$

which is a valid SVD of $\boldsymbol{A}$. Assuming $\tilde{r}\leq\min(m,n)$, the computational complexity is dominated by the QR-decompositions and scales as $\mathcal{O}((m+n)\tilde{r}^2)$.

Rank Truncation Performing a rank truncation to a smaller rank $r'<r$ on $\boldsymbol{A}$ is trivial, as it is sufficient to truncate $\boldsymbol{U}$ and $\boldsymbol{V}$ to r' columns and $\boldsymbol{\Sigma}$ to r' rows and columns. By Theorem 3.5, this is the best rank r' approximation of $\boldsymbol{A}$. The computational complexity is determined solely by the copying of the values and scales as $\mathcal{O}(r'(m+n))$, compared to $\mathcal{O}(mn\min(m,n))$ for directly stored matrices.

Access Entries A drawback of the decomposed representation is that the entries $\boldsymbol{A}[i,j]$ cannot be accessed directly. Instead, one has to compute

$$\boldsymbol{A}[i,j] = \left(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\right)[i,j] = \sum_{k=1}^{r} \boldsymbol{U}[i,k]\boldsymbol{\Sigma}[k,k]\boldsymbol{V}[j,k] \ .$$

The computation cost scales as $\mathcal{O}(r)$, compared to the constant access cost for directly stored matrices.

Addition Computing the sum $\boldsymbol{A} + \bar{\boldsymbol{A}}$ is possible while remaining in a low rank representation at all times. To this end, construct

$$\begin{aligned} \boldsymbol{W}_L &= \begin{pmatrix} \boldsymbol{U} & \bar{\boldsymbol{U}} \end{pmatrix} \in \mathbb{R}^{m \times r+\bar{r}} \\ \boldsymbol{\Sigma} &= \begin{pmatrix} \boldsymbol{\Sigma} & \boldsymbol{0} \\ 0 & \bar{\boldsymbol{\Sigma}} \end{pmatrix} \in \mathbb{R}^{r+\bar{r} \times r+\bar{r}} \\ \boldsymbol{W}_R^T &= \begin{pmatrix} \boldsymbol{V}^T \\ \bar{\boldsymbol{V}}^T \end{pmatrix} \in \mathbb{R}^{r+\bar{r} \times n} \end{aligned}$$

It is easy to verify that

$$\boldsymbol{W}_L \boldsymbol{\Sigma} \boldsymbol{W}_R^T = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T + \bar{\boldsymbol{U}}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{V}}^T = \boldsymbol{A} + \bar{\boldsymbol{A}}$$

holds, which is therefore a rank $r + \bar{r}$ representation of $\boldsymbol{A}$. However, note that $\boldsymbol{A}$ is in general not of rank $r + \bar{r}$ and $\boldsymbol{W}_L, \boldsymbol{W}_R$ are not necessarily orthogonal. Therefore, in order to stay in the SVD format, a subsequent reorthogonalization is necessary. The computational costs for the addition itself scale as $\mathcal{O}((m+n)(r+\bar{r}))$, compared to $\mathcal{O}(mn)$ for directly stored matrices.

Hadamard Product The Hadamard or entrywise product $\boldsymbol{A} \odot \bar{\boldsymbol{A}}$ can be calculated more efficiently than for directly stored matrices. For this, let $u_{ij} := \boldsymbol{U}[i,j]$ and $\bar{u}_{ij} := \bar{\boldsymbol{U}}[i,j]$ denote the entries of $\boldsymbol{U}$ and $\bar{\boldsymbol{U}}$, analogous for $v_{ij} := \boldsymbol{V}[i,j]$ and $\bar{v}_{ij} := \bar{\boldsymbol{V}}[i,j]$. Denote the singular values as $\sigma_i := \boldsymbol{\Sigma}[i,i]$ and $\bar{\sigma}_i := \bar{\boldsymbol{\Sigma}}[i,i]$. Define the matrices

$$\begin{aligned} \boldsymbol{W}_L &:= \begin{pmatrix} u_{11}\bar{u}_{11} & u_{11}\bar{u}_{12} & \dots & u_{11}\bar{u}_{1\bar{r}} & u_{12}\bar{u}_{11} & \dots & u_{1r}\bar{u}_{1\bar{r}} \\ u_{21}\bar{u}_{21} & u_{21}\bar{u}_{22} & \dots & u_{21}\bar{u}_{2\bar{r}} & u_{22}\bar{u}_{21} & \dots & u_{2r}\bar{u}_{2\bar{r}} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ u_{m1}\bar{u}_{m1} & u_{m1}\bar{u}_{m2} & \dots & u_{m1}\bar{u}_{m\bar{r}} & u_{m2}\bar{u}_{m1} & \dots & u_{mr}\bar{u}_{m\bar{r}} \end{pmatrix} \in \mathbb{R}^{m \times r\bar{r}} \\ \boldsymbol{D} &:= \operatorname{diag}(\sigma_1\bar{\sigma}_1, \sigma_1\bar{\sigma}_2, \dots, \sigma_1\bar{\sigma}_{\bar{r}}, \sigma_2\bar{\sigma}_1, \dots, \sigma_r\bar{\sigma}_{\bar{r}}) \in \mathbb{R}^{r\bar{r} \times r\bar{r}} \\ \boldsymbol{W}_R &:= \begin{pmatrix} v_{11}\bar{v}_{11} & v_{11}\bar{v}_{12} & \dots & v_{11}\bar{v}_{1\bar{r}} & v_{12}\bar{v}_{11} & \dots & v_{1r}\bar{v}_{1\bar{r}} \\ v_{21}\bar{v}_{21} & v_{21}\bar{v}_{22} & \dots & v_{21}\bar{v}_{2\bar{r}} & v_{22}\bar{v}_{21} & \dots & v_{2r}\bar{v}_{2\bar{r}} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ v_{m1}\bar{v}_{m1} & v_{m1}\bar{v}_{m2} & \dots & v_{m1}\bar{v}_{m\bar{r}} & v_{m2}\bar{v}_{m1} & \dots & v_{mr}\bar{v}_{m\bar{r}} \end{pmatrix} \in \mathbb{R}^{n \times r\bar{r}} \end{aligned}$$

Then,

$$\begin{aligned}\left(\boldsymbol{W}_L\boldsymbol{D}\boldsymbol{W}_R^T\right)[i,j] &= \sum_{k=1}^{r\bar{r}} \boldsymbol{W}_L[i,k]\boldsymbol{D}[k,k]\boldsymbol{W}_R[j,k] \\ &= \sum_{p=1}^{r}\sum_{q=1}^{\bar{r}} \boldsymbol{W}_L[i,p\bar{r}+q]\boldsymbol{D}[p\bar{r}+q,p\bar{r}+q]\boldsymbol{W}_R[j,p\bar{r}+q] \\ &= \sum_{p=1}^{r}\sum_{q=1}^{\bar{r}} u_{ip}\bar{u}_{iq}\sigma_p\bar{\sigma}_q v_{jp}\bar{v}_{jq} \\ &= \sum_{p=1}^{r} u_{ip}\sigma_p v_{jp}\sum_{q=1}^{\bar{r}} \bar{u}_{iq}\bar{\sigma}_q\bar{v}_{jq} \\ &= \left(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\right)[i,j]\left(\bar{\boldsymbol{U}}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{V}}^T\right)[i,j] \\ &= \left(\boldsymbol{A}\odot\bar{\boldsymbol{A}}\right)[i,j]\end{aligned}$$

is a rank $r\bar{r}$ representation of the Hadamard product. The computational costs to create the matrices $\boldsymbol{W}_L, \boldsymbol{D}$ and $\boldsymbol{W}_R$ scale as $\mathcal{O}((m+n)r\bar{r})$. Note, however, that to obtain a valid SVD representation, a subsequent reorthogonalization is necessary.

Frobenius Inner Product The Frobenius inner product can be calculated as follows:

$$\begin{aligned}\left\langle \boldsymbol{A}, \bar{\boldsymbol{A}}\right\rangle_F &= \left\langle \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \bar{\boldsymbol{U}}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{V}}^T\right\rangle_F \\ &= \left\langle \bar{\boldsymbol{U}}^T\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\bar{\boldsymbol{V}}, \bar{\boldsymbol{\Sigma}}\right\rangle_F \\ &= \left\langle \boldsymbol{W}_L\boldsymbol{\Sigma}\boldsymbol{W}_R, \bar{\boldsymbol{\Sigma}}\right\rangle_F \ .\end{aligned}$$

The computational complexity is determined by the computation of $\boldsymbol{W}_L$ and $\boldsymbol{W}_R$ and scales as $\mathcal{O}((m+n)r\bar{r})$, compared to $\mathcal{O}(nm)$ for directly stored matrices.

Frobenius Norm As the application of an orthogonal matrix does not change the norm, it holds that

$$\|\boldsymbol{A}\|_F = \|\boldsymbol{\Sigma}\|_F = \sqrt{\sum_{k=1}^{r}\sigma_k^2(\boldsymbol{A})} \ .$$

This has a computational complexity of only $\mathcal{O}(r)$, compared to $\mathcal{O}(nm)$ for directly stored matrices.

Matrix Product Let $\boldsymbol{B} \in \mathbb{R}^{n\times p}$ be a matrix of rank r', given as the SVD $\boldsymbol{B} = \boldsymbol{U}'\boldsymbol{\Sigma}'\boldsymbol{V}'^T$. The matrix-matrix product $\boldsymbol{AB}$ can be computed inexpensively as

$$\boldsymbol{AB} = \boldsymbol{U}\underbrace{\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{U}'\boldsymbol{\Sigma}'}_{:=\boldsymbol{C}}\boldsymbol{V}'^T = \boldsymbol{U}\boldsymbol{C}\boldsymbol{V}'^T \ .$$

Calculating a SVD of $\boldsymbol{U}_c\boldsymbol{D}\boldsymbol{V}_c^T = \boldsymbol{C} \in \mathbb{R}^{r\times r'}$ gives

$$\boldsymbol{AB} = \boldsymbol{U}\boldsymbol{C}\boldsymbol{V}'^T = \underbrace{\boldsymbol{U}\boldsymbol{U}_c}_{:=\boldsymbol{W}_L}\boldsymbol{D}\underbrace{\boldsymbol{V}_c^T\boldsymbol{V}'^T}_{:=\boldsymbol{W}_R^T} = \boldsymbol{W}_L\boldsymbol{D}\boldsymbol{W}_R^T \ ,$$

which is a valid SVD of the product $\boldsymbol{AB}$. The computational costs are dominated by the computation of $\boldsymbol{C}, \boldsymbol{W}_L, \boldsymbol{W}_R$. The scaling of the costs is bounded by $\mathcal{O}((m+n+p)rr')$, compared to $\mathcal{O}(mnp)$ for directly stored matrices.

3.2 Canonical Polyadic Decomposition

The canonical rank and the corresponding canonical polyadic (CP) decomposition is likely the most widely known generalization of the matrix rank and matrix SVD to higher order tensors. It is usually attributed to Hitchcock [14], who published his work in 1927. In the literature, the names *PARAFAC* (Parallel factor analysis) and *Candecomp* (Canonical decomposition) are also commonly used.

3.2.1 Definition

The rank generalization of the CP decomposition is based on Statement V in Definition 3.3, which defines the rank of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ as the minimal $r \in \mathbb{N}$, such that $\boldsymbol{A}$ can be represented as a sum of r dyadic vector products

$$\boldsymbol{A} = \sum_{k=1}^{r} \boldsymbol{v}_k \boldsymbol{u}_k^T = \sum_{k=1}^{r} \boldsymbol{v}_k \otimes \boldsymbol{u}_k \qquad\qquad \boldsymbol{v}_k \in \mathbb{R}^m, \boldsymbol{u}_k \in \mathbb{R}^n \ .$$

This is straightforwardly generalized to higher order tensors, as formalized in the following definition.

Definition 3.10 (Canonical Polyadic Decomposition)**.** Let $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ be a tensor of order d. A representation of $\mathcal{X}$ as a sum of elementary tensors

$$\mathcal{X} = \sum_{p=1}^{r} \boldsymbol{v}_{1,p} \otimes \ldots \otimes \boldsymbol{v}_{d,p} = \sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} \qquad\qquad \boldsymbol{v}_{\mu,p} \in \mathbb{R}^{n_\mu} \tag{3.3}$$

is called a canonical polyadic (CP) representation of $\mathcal{X}$. The number of terms r is called the rank of the representation. The minimal r, such that there exists a CP decomposition of $\mathcal{X}$ with rank r, is called the *canonical rank* or *CP-rank* of $\mathcal{X}$.

By defining matrices $\boldsymbol{V}_\mu = (\boldsymbol{v}_{\mu,1}, \ldots, \boldsymbol{v}_{\mu,r}) \in \mathbb{R}^{n_\mu \times r}$, (3.3) can equivalently be expressed as an entrywise equation

$$\mathcal{X}[i_1, \ldots, i_d] = \sum_{p=1}^{r} \boldsymbol{V}_1[i_1, p] \boldsymbol{V}_2[i_2, p] \ldots \boldsymbol{V}_d[i_d, p] \ .$$

Note that for matrices, i.e. $d = 2$, the canonical rank naturally coincides with the matrix rank and the canonical decomposition is the same as in Statement V of Definition 3.3. In order for the canonical rank to also be well-defined for higher order tensors, it has to be shown that every tensor admits at least one CP representation. This is indeed the case and follows directly from Proposition 2.9. However, note that in general there is no *unique* CP representation with minimal rank. This is somewhat expected, since even for matrices the SVD is not unique if two or more singular values coincide. Some further discussion on the uniqueness for higher order tensors can be found in the survey by Kolda and Bader [45].

3.2.2 Calculation of a Decomposition

Unfortunately, the CP format suffers from a number of theoretical and numerical drawbacks. One main issue is that there is no equivalent of the numerical matrix SVD for the canonical format. In fact, even determining the canonical rank of a tensor of order $d > 2$ is, in contrast to matrices, a fundamentally open problem. It was shown by Håstad [78] that even for order $d = 3$, the problem of deciding whether a rational tensor has CP-rank r is NP-hard (and NP-complete for finite fields). The situation is in some respects even worse when trying to approximate a given tensor $\mathcal{X}$ by a tensor $\mathcal{X}_r$ with CP-rank at most r. In particular, one is usually interested in the best CP-rank r approximation, that is,

$$\mathcal{X}^* = \underset{\text{CP-rank}(\mathcal{X}_r)\leq r}{\operatorname{argmin}} \left(\|\mathcal{X} - \mathcal{X}_r\|\right) \ . \tag{3.4}$$

The norm $\|\cdot\|$ used may differ depending on the application. In the matrix case, the Eckart-Young theorem (Theorem 3.5) shows that for the Frobenius and spectral norm, this best approximation can be calculated directly by a truncated SVD. In contrast, De Silva and Lim [79] proved that the problem of the best CP-rank r approximation, as formulated in (3.4), is ill-posed for many ranks $r \geq 2$ and all orders $d \geq 3$, regardless of the choice of the norm $\|\cdot\|$. Furthermore, they showed that the set of tensors that fail to have a best CP-rank r approximation is a non-null set, i.e. there is a strictly positive probability that a randomly chosen tensor does not even admit a best CP-rank r approximation.

Nevertheless, there are methods for the calculation of approximate CP decompositions and approximations of higher order tensors. Usually, this challenging and expensive task is approached using optimization algorithms, see for example the survey by Kolda and Bader [45] and the numerical comparison of several algorithms by Faber et al. [80].

3.2.3 The Set of Tensors with Fixed Canonical Rank

The problems in finding a best rank r approximation in the CP format is tightly linked to the fact that, as shown by De Silva and Lim [79], neither the set $\{\mathcal{X} \in \mathbb{R}^{n_1\times\ldots\times n_d} \mid \text{CP-rank}(\mathcal{X}) = r\}$ of all tensors with CP-rank r, nor the set $\{\mathcal{X} \in \mathbb{R}^{n_1\times\ldots\times n_d} \mid \text{CP-rank}(\mathcal{X}) \leq r\}$ of all tensors with CP-rank at most r, are closed for $d > 2$. In particular, this means that a sequence of higher order tensors with CP-rank (at most) r may converge to a tensor with greater rank. A popular example by De Silva and Lim [79] is the sequence of tensors

$$\mathcal{X}_n = n\left(\boldsymbol{u} + \frac{1}{n}\boldsymbol{v}\right) \otimes \left(\boldsymbol{u} + \frac{1}{n}\boldsymbol{v}\right) \otimes \left(\boldsymbol{u} + \frac{1}{n}\boldsymbol{v}\right) - n\boldsymbol{u} \otimes \boldsymbol{u} \otimes \boldsymbol{u} \ ,$$

with $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^m$, $\|\boldsymbol{u}\| = \|\boldsymbol{v}\| = 1$ and $\langle \boldsymbol{u}, \boldsymbol{v} \rangle \neq 1$. Here, each $\mathcal{X}_n$ is given in a CP representation with representation rank two, hence the canonical rank of each $\mathcal{X}_n$ is at most two.[2] The limit of this sequence is the tensor

$$\mathcal{X} = \boldsymbol{v} \otimes \boldsymbol{u} \otimes \boldsymbol{u} + \boldsymbol{u} \otimes \boldsymbol{v} \otimes \boldsymbol{u} + \boldsymbol{u} \otimes \boldsymbol{u} \otimes \boldsymbol{v} \ ,$$

which is of canonical rank three.

The fact that neither set of low rank tensors is closed, poses both a theoretical and practical problem. Additionally, in contrast to Proposition 3.9, which shows that the set of fixed rank matrices allows a manifold structure, there is no known equivalent structure for the set $\{\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d} \mid \text{CP-rank}(\mathcal{X}) = r\}$ of tensors with CP-rank r. In particular, this severely complicates the treatment of optimization algorithms operating on these sets. This is also due to the fact that there is only very limited normalization available for the CP-format, and the representation in (3.3) is usually ambiguous, in the sense that there are many different sets of parameters $\boldsymbol{v}_{\mu,p}$ which approximate the same tensor. The work of Mitchell and Burdick [81] presents several examples of tensors, which are difficult to optimize.

3.2.4 Computational Aspects

Apart from the difficulties described in the previous two sections, the CP format allows an unparalleled complexity reduction for tensors with small canonical rank. As shown in Chapter 2, general tensors suffer from the *curse of dimensionality*, that is, the storage complexity, as well as the computational complexity of common operations, scale exponentially in the order. In particular, given a tensor space $\mathbb{R}^{n_1 \times \dots \times n_d}$ and $n = \max_\mu(n_\mu)$, storing a general tensor requires $\Theta(n^d)$ storage space. In contrast, for fixed CP-rank r, the canonical representation only needs to store the entries of the vectors $\boldsymbol{v}_{\mu,p}$ in (3.3). This requires only $\Theta(dnr)$ storage, thus scaling only *linearly* in the order. As shown in the following, other common operations can be performed in non-exponentially scaling complexity as well if all operands are given in a low rank CP representation. The result of the operation is directly obtained in a low rank representation as well. Accordingly, the CP-format is said to break the curse of dimensionality, in the sense that no exponentially scaling complexity occurs when working solely in the CP-format with tensors of bounded CP-rank.

The following list of operations is analogous to Section 3.1.5 for matrices. For the remainder of this section, let $\mathcal{X}, \bar{\mathcal{X}} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ be two tensors with CP-rank r and $\bar{r}$, respectively, given in CP representations $\mathcal{X} = \sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p}$ and $\bar{\mathcal{X}} = \sum_{q=1}^{\bar{r}} \bigotimes_{\mu=1}^{d} \bar{\boldsymbol{v}}_{\mu,q}$.

[2] They actually have canonical rank two, as can be shown for example by the fact that the matricization $\boldsymbol{X}_n^{(1)}$ has rank two, as will be clear at the end of this chapter.

Normalization and Rank Truncation As mentioned in the previous sections, there are no straightforward methods to determine the canonical rank or to compute a low CP rank representation. This situation remains unchanged even if a (sub-optimal) low CP rank representation is available. For most of the following operations, this has the inconvenient consequence that no simple rank reduction can be performed on the results, which are usually of sub-optimal rank. Additionally, the CP-format also misses much of the normalization/orthogonalization of the matrix SVD. In particular, in the matrix representation in Statement V in Definition 3.3, the vectors $\boldsymbol{u}_i, \boldsymbol{v}_i$ can be chosen to be orthonormal, yielding an almost unique representation.[3] This is not the case for the CP-format, e.g. requiring all $\boldsymbol{v}_{\mu,p}$ in (3.3) to be orthogonal for each μ changes the rank definition, as tensors with CP-rank r do not necessarily allow such an orthogonal representation with the same rank, as explained in the work of Kolda [82].

Access Entries As for matrices, the entries $\mathcal{X}[i_1, \dots, i_d]$ cannot be accessed directly in the CP-representation. Instead, one has to compute

$$\mathcal{X}[i_1, \dots, i_d] = \left(\sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} \right) [i_1, \dots, i_d] = \sum_{p=1}^{r} \prod_{\mu=1}^{d} \boldsymbol{v}_{\mu,p}[i_\mu] \ .$$

The computational cost scales as $\mathcal{O}(dr)$, compared to the constant access cost for a directly stored tensor.

Addition The addition of $\mathcal{X}$ and $\bar{\mathcal{X}}$ can be trivially computed by combining the sums, i.e.

$$\mathcal{X} + \bar{\mathcal{X}} = \sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} + \sum_{q=1}^{\bar{r}} \bigotimes_{\mu=1}^{d} \bar{\boldsymbol{v}}_{\mu,q} = \sum_{k=1}^{r+\bar{r}} \bigotimes_{\mu=1}^{d} \boldsymbol{w}_{\mu,k} \ ,$$

with

$$\boldsymbol{w}_{\mu,k} := \begin{cases} \boldsymbol{v}_{\mu,k} & k \leq r \\ \bar{\boldsymbol{v}}_{\mu,k-r} & k > r \end{cases} \ .$$

This is a CP representation of the sum with rank $r + \bar{r}$, however, the actual CP rank of $\mathcal{X} + \bar{\mathcal{X}}$ can be smaller. The only contribution to the computational complexity is the copying of the vectors $\boldsymbol{w}_{\mu,k}$ which scales as $\mathcal{O}(dn(r + \bar{r}))$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

Hadamard Product The Hadamard product can be calculated within the canonical representation, by using

$$\mathcal{X} \odot \bar{\mathcal{X}} = \left(\sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} \right) \left(\sum_{q=1}^{\bar{r}} \bigotimes_{\mu=1}^{d} \bar{\boldsymbol{v}}_{\mu,q} \right) = \sum_{p=1}^{r} \sum_{q=1}^{\bar{r}} \bigotimes_{\mu=1}^{d} (\boldsymbol{v}_{\mu,p} \odot \bar{\boldsymbol{v}}_{\mu,q}) \ .$$

[3]Phase factors can still be applied and if two or more singular values coincide, an arbitrary orthonormal basis for the corresponding spaces can be chosen.

This gives a rank $r + \bar{r}$ canonical representation of the Hadamard product. The computational costs are caused by the calculation of the entrywise vector products and amount to $\mathcal{O}(dnr\bar{r})$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

Frobenius Inner Product Using the fact that the Frobenius scalar product is the induced scalar product (see Definition 2.10), the inner product $\langle \mathcal{X}, \bar{\mathcal{X}} \rangle$ can be computed as

$$\begin{aligned}\left\langle \mathcal{X}, \bar{\mathcal{X}} \right\rangle_F &= \left\langle \sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p},\ \sum_{q=1}^{\bar{r}} \bigotimes_{\mu=1}^{d} \bar{\boldsymbol{v}}_{\mu,q} \right\rangle_F \\ &= \sum_{p=1}^{r} \sum_{q=1}^{\bar{r}} \left\langle \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p},\ \bigotimes_{\mu=1}^{d} \bar{\boldsymbol{v}}_{\mu,q} \right\rangle_F \\ &= \sum_{p=1}^{r} \sum_{q=1}^{\bar{r}} \prod_{\mu=1}^{d} \langle \boldsymbol{v}_{\mu,p}, \bar{\boldsymbol{v}}_{\mu,q} \rangle \ .\end{aligned}$$

The computational complexity scales as $\mathcal{O}(dnr\bar{r})$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

Frobenius Norm Using the above result, the Frobenius norm of $\mathcal{X}$ can be computed as

$$\|\mathcal{X}\|_F = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle_F} = \sqrt{\sum_{p=1}^{r_1} \sum_{q=1}^{r_1} \prod_{\mu=1}^{d} \langle \boldsymbol{v}_{\mu,p}, \boldsymbol{v}_{\mu,q} \rangle} \ .$$

The computational complexity scales as $\mathcal{O}(dnr^2)$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

ν-th Mode Product Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n_\nu \times m}$, the ν-th mode product $\mathcal{X} \times_\nu \boldsymbol{A}$ can be computed as

$$\begin{aligned}\mathcal{X} \times_\nu \boldsymbol{A} &= \left(\sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} \right) \times_\nu \boldsymbol{A} \\ &= \sum_{p=1}^{r} \left(\bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} \right) \times_\nu \boldsymbol{A} \\ &= \sum_{p=1}^{r} \boldsymbol{v}_{1,p} \otimes \ldots \otimes \left(\boldsymbol{A}^T \boldsymbol{v}_{\nu,p} \right) \otimes \ldots \otimes \boldsymbol{v}_{d,p} \ .\end{aligned}$$

This is a CP representation of the same rank r. The computational cost consists solely of the copying and the r matrix-vector products, it thus scales as $\mathcal{O}(dnr + mnr)$, compared to $\mathcal{O}(n^d m)$ for directly stored tensors.

Other Contractions General contractions can be computed while staying in a low rank representation at all times, if both operands are given in a CP representation. The following shows this for the contraction of a single mode pair. The contraction of multiple mode pairs can be shown analogously.

Let $\mathcal{Y} \in \mathbb{R}^{m_1 \times \dots \times m_e}$ be another tensors of order e given in a CP decomposition $\mathcal{Y} = \sum_{q=1}^{\bar{r}} \bigotimes_{\nu=1}^{e} \boldsymbol{u}_{\nu,q}$. Assuming $m_\tau = n_\rho$ holds, the contraction $\mathcal{Y} *_{\tau,\rho} \mathcal{X}$ of the τ-th mode of $\mathcal{Y}$ with the ρ-th mode of $\mathcal{X}$ can be calculated as

$$\begin{aligned}
\mathcal{Y} *_{\tau,\rho} \mathcal{X} &= \left(\sum_{q=1}^{\bar{r}} \bigotimes_{\nu=1}^{e} \boldsymbol{u}_{\nu,q} \right) *_{\tau,\rho} \left(\sum_{p=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} \right) \\
&= \sum_{q=1}^{\bar{r}} \sum_{p=1}^{r} \left(\bigotimes_{\nu=1}^{e} \boldsymbol{u}_{\nu,q} \right) *_{\tau,\rho} \left(\bigotimes_{\mu=1}^{d} \boldsymbol{v}_{\mu,p} \right) \\
&= \sum_{p=1}^{r} \sum_{q=1}^{\bar{r}} \langle \boldsymbol{u}_{\tau,q}, \boldsymbol{v}_{\rho,p} \rangle \bigotimes_{\substack{\nu=1 \\ \nu \neq \tau}}^{e} \boldsymbol{u}_{\nu,q} \otimes \bigotimes_{\substack{\mu=1 \\ \mu \neq \rho}}^{d} \boldsymbol{v}_{\mu,p} \ .
\end{aligned}$$

Absorbing the scalar $\langle \boldsymbol{u}_{\tau,q}, \boldsymbol{v}_{\rho,p} \rangle$ in $\boldsymbol{u}_{1,q}$ in each term, this gives a rank $r\bar{r}$ CP representation of the contraction. The computation cost is dominated by the copy operations and scales as $\mathcal{O}((d+e)r\bar{r}\max(m,n))$, compared to $\mathcal{O}(m_\tau m^{d-1} n^{d-1})$ for directly stored tensors.

3.3 Tucker Decomposition

The Tucker decomposition is a subspace based approach, which is able to avoid some of the shortcomings of the canonical format. It is usually attributed to Tucker [15], who described a special case of the decomposition in 1966. The formulation of the Tucker decomposition as a generalization of the matrix SVD and, in particular, the higher order SVD introduced in this section are due to De Lathauwer et al. [83].

The fundamental idea of the Tucker decomposition is to generalize the matrix rank using Item VI in Definition 3.3, which states that the rank r of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is equivalently defined as the smallest number r, such that there exist r-dimensional subspaces $V \subseteq \mathbb{R}^m$ and $U \subseteq \mathbb{R}^n$, such that $\boldsymbol{A}$ is an element of the induced tensor space $V \otimes U \subseteq \mathbb{R}^{m \times n}$. As elaborated in Section 3.1.1, for matrices, these minimal subspaces are $V = \operatorname{im}(\boldsymbol{A})$ and $U = \mathbb{R}^m / \ker(\boldsymbol{A})$, and their dimension always coincide. Generalizing this to higher order tensors, the task is to find minimal subspaces, such that the given tensor is an element of the induced tensor space. Importantly, in contrast to the matrix case, the dimensions of these minimal subspaces are not necessarily all equal. Therefore, instead of a single number, the Tucker rank is defined as the tuple of these dimensions.

3.3.1 Definition

For any tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, let $U_\mu \subseteq \mathbb{R}^{n_\mu}$ be a, not necessarily minimal, set of subspaces, such that $\mathcal{X}$ is an element of the induced tensor space $\bigotimes_{\mu=1}^{d} U_\mu$. Let $r_\mu = \dim(U_\mu)$ denote the dimension of the μ-th subspace and let $\{\boldsymbol{u}_{\mu,p_\mu} \mid p_\mu = 0, \dots, r_\mu\}$ be an orthonormal basis of U_μ, for all μ. As $\mathcal{X} \in \bigotimes_{\mu=1}^d U_\mu$, it can by definition be expressed as a linear combination of elementary tensors, composed of the basis vectors $\boldsymbol{u}_{\mu,p_\mu}$. In particular, it can always be written as a linear combination of *all* elementary tensors composed of these basis vectors

$$\mathcal{X} = \sum_{p_1=1}^{r_1} \cdots \sum_{p_d=1}^{r_d} \mathcal{C}[p_1, \dots, p_d] \cdot \boldsymbol{u}_{1,p_1} \otimes \dots \otimes \boldsymbol{u}_{d,p_d} \ ,$$

with some pre-factors $\mathcal{C}[p_1, \dots, p_d]$ possibly being zero. As indicated by the notation, the prefactors will themselves be interpreted as an order d tensor $\mathcal{C} \in \mathbb{R}^{r_1 \times \dots \times r_d}$, the so-called *core tensor*. The sets of basis vectors can also be viewed as matrices $\boldsymbol{U}_\mu = (\boldsymbol{u}_{\mu,1}, \dots, \boldsymbol{u}_{\mu,r_\mu})^T$, called *basis matrices*, given in entrywise notation by $\boldsymbol{U}_\mu[p_\mu, k] := \boldsymbol{u}_{\mu,p_\mu}[k]$. As the basis vectors are chosen to be orthonormal, the basis matrices are left-orthogonal. Using this matrix representation of the basis, the Tucker rank and the Tucker format are formally defined as follows.

Definition 3.11 (Tucker Decomposition)**.** Let $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ be a tensor of order d. A Tucker representation of $\mathcal{X}$ is given by right-orthogonal matrices $\boldsymbol{U}_\mu \in \mathbb{R}^{r_\mu \times n_\mu}$ and a core tensor $\mathcal{C} \in \mathbb{R}^{r_1 \times \dots \times r_d}$,[4] such that

$$\mathcal{X} = \mathcal{C} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \dots \times_d \boldsymbol{U}_d \ , \tag{3.5}$$

or, in entrywise notation,

$$\mathcal{X}[i_1, \dots, i_d] = \sum_{p_1, \dots, p_d} \mathcal{C}[p_1, \dots, p_d] \boldsymbol{U}_1[p_1, i_1] \boldsymbol{U}_1[p_2, i_2] \dots \boldsymbol{U}_d[p_d, i_d] \ .$$

The d-tuple $(r_1, r_2, \dots, r_d)$ is called the representation rank and is associated with the particular representation. In contrast, the *Tucker rank* (T-rank) of $\mathcal{X}$ is defined as the minimal d-tuple $\boldsymbol{r} = (r_1, \dots, r_d)$, such that there exists a Tucker representation of $\mathcal{X}$ with rank $\boldsymbol{r}$.

The Tucker decomposition (3.5) consists solely of multiple tensor contractions. Accordingly, the Tucker decomposition can also be interpreted as a representation of a given tensor by a tensor network with a specific topology. An example of such a network topology is visualized in Figure 3.1 for an order six tensor.

It is clear that every tensor admits a Tucker decomposition, since choosing all subspaces $U_\mu = \mathbb{R}^{n_\mu}$ equal to the original spaces and using an arbitrary orthogonal basis always yields a valid Tucker decomposition. On the other hand, it is not obvious *a priori* that the Tucker rank as the minimal d-tuple is well-defined. In fact, in order to talk of a minimal rank at all, some kind of order on the set of d-tuples is needed. In Definition 3.11 and in the remainder of this thesis, the following partial order is used for all m-tuples.

[4]It is possible to impose further restrictions on the core tensor $\mathcal{C}$, see for example the work of De Lathauwer et al. [83]. However, these restrictions have no impact on the rank definition and all results of this thesis are compatible with these restrictions through some obvious modifications.

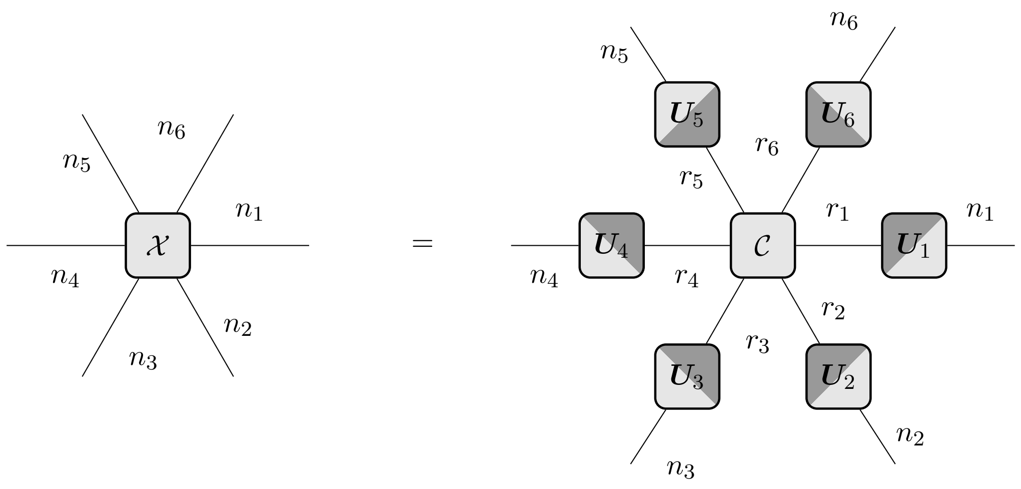


Figure 3.1: Left: A general tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ of order six. Right: Its Tucker decomposition.

Definition 3.12 (Partial order of m-tuples)**.** On the set of m-tuples, the partial order $\preceq$ is defined for all $\boldsymbol{x} = (x_1, \dots, x_m)$ and $\boldsymbol{y} = (y_1, \dots, y_m)$ as

$$(x_1, \dots, x_m) \preceq (y_1, \dots, y_m) \;\Leftrightarrow\; \forall i: \; x_i \leq y_i \;.$$

The relation $\succeq$ is naturally defined as $\boldsymbol{x} \succeq \boldsymbol{y} :\Leftrightarrow \boldsymbol{y} \preceq \boldsymbol{x}$ and so are $\boldsymbol{x} \prec \boldsymbol{y} :\Leftrightarrow \boldsymbol{x} \preceq \boldsymbol{y}, \boldsymbol{x} \neq \boldsymbol{y}$ and $\boldsymbol{x} \succ \boldsymbol{y} :\Leftrightarrow \boldsymbol{x} \succeq \boldsymbol{y}, \boldsymbol{x} \neq \boldsymbol{y}$. Furthermore, let X be a set of m-tuples. Then, $\boldsymbol{y} = (y_1, \dots, y_m)$ is the smallest element of X, iff for all $\boldsymbol{x} \in X$, it holds that $\boldsymbol{y} \preceq \boldsymbol{x}$. Note that the smallest element might not exist, which therefore has to be checked separately. However, if there is a smallest element, it is unique.

In the following, expressions such as "the minimal rank" always refer to the smallest element in this sense. In order to prove that the Tucker rank is well defined, it has to be shown that there exists a minimal d-tuple for which a Tucker representation exists. As will be shown in the following section, this is indeed the case and the minimal rank is linked to ranks of certain matricizations. The proof is constructive and directly gives an algorithm to compute the Tucker decomposition.

3.3.2 Calculation of a Decomposition

It is shown by De Lathauwer et al. [83] that a Tucker decomposition with minimal representation rank can be obtained by successive matrix SVDs. This procedure is commonly referred to as the *higher order singular value decomposition* (HOSVD) and is formalized in Algorithm 1. The idea is to calculate the minimal subspaces in form of the basis matrices $\boldsymbol{U}_\mu$ via singular value decompositions of the single mode matricizations of the tensor. Subsequently, the core tensor is obtained by projection onto these subspaces. The following theorem shows that this decomposition is exact and indeed of minimal representation rank.

Algorithm 1: Higher Order Singular Value Decomposition

```
Input   : Target $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$
Output : Core tensor $\mathcal{C} \in \mathbb{R}^{r_1 \times \dots , r_d}$, basis matrices $\boldsymbol{U}_1 \in \mathbb{R}^{r_1 \times n_1}, \dots, \boldsymbol{U}_d \in \mathbb{R}^{r_d \times n_d}$
1 for $\mu = 1, \dots, d$ do
2     Calculate $\tilde{\boldsymbol{U}}_\mu \boldsymbol{S}_\mu \boldsymbol{V}_\mu^T := \text{SVD}\Big(\boldsymbol{X}^{(\mu)}\Big)$
3     Set $\boldsymbol{U}_\mu := \tilde{\boldsymbol{U}}_\mu^T$
4 Calculate $\mathcal{C} := \mathcal{X} \times_1 \boldsymbol{U}_1^T \times_2 \boldsymbol{U}_2^T \dots \times_d \boldsymbol{U}_d^T$
```

Theorem 3.13. *Given a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, let $\mathcal{C}$ be the core tensor and $\boldsymbol{U}_1, \dots, \boldsymbol{U}_d$ the basis matrices obtained by the HOSVD in Algorithm 1. Then,*

$$\mathcal{X} = \mathcal{C} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \dots \times_d \boldsymbol{U}_d \tag{3.6}$$

is a Tucker representation of $\mathcal{X}$ of minimal rank. The obtained representation rank, which thereby is also the Tucker rank of $\mathcal{X}$, is related to the matrix rank of the μ-th mode matricizations via

$$\text{T-rank}(\mathcal{X}) = \Big(\text{rank}(\boldsymbol{X}^{(1)}), \dots, \text{rank}(\boldsymbol{X}^{(d)})\Big) \ . \tag{3.7}$$

Proof. We first show that (3.6) holds. From the SVD-based definition of $\boldsymbol{U}_\mu$, it is clear that for all μ, $\boldsymbol{U}_\mu^T \boldsymbol{U}_\mu \boldsymbol{X}^{(\mu)} = \boldsymbol{X}^{(\mu)}$ holds, which is equivalent to $\mathcal{X} \times_\mu \boldsymbol{U}_\mu^T \boldsymbol{U}_\mu = \mathcal{X}$. Therefore, it follows that

$$\begin{aligned} \mathcal{C} \times_1 \boldsymbol{U}_1 \dots \times_d \boldsymbol{U}_d &= \Big(\mathcal{X} \times_1 \boldsymbol{U}_1^T \dots \times_d \boldsymbol{U}_d^T\Big) \times_1 \boldsymbol{U}_1 \dots \times_d \boldsymbol{U}_d \\ &= \mathcal{X} \times_1 \boldsymbol{U}_1^T \boldsymbol{U}_1 \dots \times_d \boldsymbol{U}_d^T \boldsymbol{U}_d \\ &= \mathcal{X} \ , \end{aligned}$$

as asserted. From the SVD-based definition of $\boldsymbol{U}_\mu$, it furthermore follows that $\boldsymbol{U}_\mu \in \mathbb{R}^{r_\mu \times n_\mu}$, where $r_\mu = \text{rank}(\boldsymbol{X}^{(\mu)})$ is the rank of the μ-th mode matricization of $\mathcal{X}$, which shows that for the representation rank (3.7) holds. It remains to show that there cannot be a Tucker decomposition with a smaller representation rank. Assume towards contradiction that

$$\mathcal{X} = \mathcal{C}' \times_1 \boldsymbol{U}_1' \times_2 \boldsymbol{U}_2' \dots \times_d \boldsymbol{U}_d'$$

is such a Tucker representation of rank $(r_1', \dots, r_d')$, where $r_\mu' < \text{rank}(\boldsymbol{X}^{(\mu)})$. With

$$\mathcal{R} := \mathcal{C}' \times_1 \boldsymbol{U}_1' \dots \times_{\mu-1} \boldsymbol{U}_{\mu-1}' \times_{\mu+1} \boldsymbol{U}_{\mu+1}' \dots \times_d \boldsymbol{U}_d' \ ,$$

we have

$$\mathcal{X} = \mathcal{C}' \times_1 \boldsymbol{U}_1' \times_2 \boldsymbol{U}_2' \dots \times_d \boldsymbol{U}_d' = \mathcal{R} \times_\mu \boldsymbol{U}_\mu' \ .$$

By assumption, $\boldsymbol{U}_\mu' \in \mathbb{R}^{r_\mu' \times n_\mu}$ and therefore,

$$\boldsymbol{X}^{(\mu)} = \boldsymbol{U}_\mu'^T \boldsymbol{R}^{(\mu)}$$

is a rank r_μ' factorization of $\boldsymbol{X}^{(\mu)}$ in contradiction to the claim that $\text{rank}(\boldsymbol{X}^{(\mu)}) > r_\mu'$. $\square$

With almost the same procedure, an approximation of $\mathcal{X}$ by a tensor $\mathcal{X}'$ with lower Tucker rank $\boldsymbol{r}' = (r'_1, \dots r'_d) \preceq (r_1, \dots, r_d)$, can be obtained. To this end, the normal SVDs in Algorithm 1 are replaced by rank r'_μ truncated SVDs. This results in a Tucker decomposition of a tensor $\mathcal{X}'$ with Tucker rank $\boldsymbol{r}'$, as demanded. Unfortunately, the Eckard-Young Theorem (Theorem 3.5) does not generalize to this HOSVD. In particular, the tensor $\mathcal{X}'$ obtained in this way is in general *not* the best rank $\boldsymbol{r}'$ approximation of $\mathcal{X}$. However, as shown by De Lathauwer et al. [83], it is a so-called quasi-best approximation. That is, the error is bounded by a factor $\sqrt{d}$ times the error of the best approximation. For many applications, this quasi-best approximation is sufficient. Finding the true best approximation of $\mathcal{X}$ is NP-hard in general, as it is shown by Hillar and Lim [84] that even finding the best rank $(1, \dots, 1)$ approximation is NP-hard.[5] In contrast to the canonical format, however, the best approximation is well defined for all ranks, since the set of tensors with Tucker rank at most $\boldsymbol{r}'$ is closed, as shown in the next section.

In practice, it is usually better to calculate the basis matrices sequentially from the partially rank reduced tensor, as shown in Algorithm 2 for the truncated HOSVD. The reason for this is a reduced computational complexity, since the tensors $\mathcal{X}_\mu$ can be much smaller than $\mathcal{X}$. The available theoretical error bounds are the same for both algorithms.

Algorithm 2: Truncated Higher Order Singular Value Decomposition

```
Input  : Target $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, maximal rank $\boldsymbol{r}$
Output : Core tensor $\mathcal{C} \in \mathbb{R}^{r_1 \times \dots, r_d}$, basis matrices $\boldsymbol{U}_1 \in \mathbb{R}^{r_1 \times n_1}, \dots, \boldsymbol{U}_d \in \mathbb{R}^{r_d \times n_d}$
1 Set $\mathcal{X}_0 := \mathcal{X}$
2 for $\mu = 1, \dots, d$ do
3     Calculate rank $r_\mu$ truncated SVD $\tilde{\boldsymbol{U}}_\mu \boldsymbol{S}_\mu \boldsymbol{V}_\mu^{(\mu)} := \mathrm{SVD}_{r_\mu}\left(\boldsymbol{X}_{\mu-1}^{(\mu)}\right)$
4     Set $\boldsymbol{X}_\mu^{(\mu)} := \boldsymbol{S}_\mu \boldsymbol{V}_\mu^{(\mu)}$
5     Set $\boldsymbol{U}_\mu := \tilde{\boldsymbol{U}}_\mu^T$
6 Set $\mathcal{C} := \mathcal{X}_d$
```

Theorem 3.14 (HOSVD Quasi Best Approximation [83])**.** *Given a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, let $\mathcal{C}$ be the core tensor and $\boldsymbol{U}_1, \dots, \boldsymbol{U}_d$ the basis matrices obtained by the rank $\boldsymbol{r}$ truncated HOSVD (Algorithm 2). These define a rank $\boldsymbol{r}$ Tucker representation*

$$\mathcal{X}' = \mathcal{C} \times_1 \boldsymbol{U}_1 \dots \times_d \boldsymbol{U}_d$$

of a tensor $\mathcal{X}'$, which is a quasi-best rank $\boldsymbol{r}$ approximation of $\mathcal{X}$, in the sense that

$$\|\mathcal{X} - \mathcal{X}'\|_F \leq \sqrt{d} \cdot \min\{\|\mathcal{X} - \mathcal{Y}\|_F \mid \mathcal{Y} \in \mathbb{R}^{n_1 \times \dots \times n_d},\ \text{T-rank}(\mathcal{Y}) \preceq \boldsymbol{r}\} \ ,$$

holds.

Proof. The original proof by De Lathauwer et al. [83] is rather technical and is omitted here. However, in Section 4.2, we use a different approach to prove the quasi-optimality of

[5]As will be shown in Section 3.6 for rank $(1, \dots, 1)$ the CP and Tucker format coincide.

the randomized HOSVD introduced in that section, which can be transformed to a proof of the present theorem with only a few obvious modifications. □

Let us conclude this subsection with the following observation, stating that the complete Frobenius norm is contained in the core tensor.

Lemma 3.15. *For every tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ and every right-orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{n_\mu \times m}$ with $\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I}$, it holds that*

$$\|\mathcal{X} \times_\mu \boldsymbol{Q}\|_F = \|\mathcal{X}\|_F \ .$$

Proof. Using the fact that matricizations do not change the Frobenius norm, it holds that

$$\begin{aligned}
\|\mathcal{X} \times_\mu \boldsymbol{Q}\|_F^2 &= \|\boldsymbol{X}^{(\mu)}\boldsymbol{Q}\|_F^2 \\
&= \left\langle \boldsymbol{X}^{(\mu)}\boldsymbol{Q}, \boldsymbol{X}^{(\mu)}\boldsymbol{Q} \right\rangle_F \\
&= \left\langle \boldsymbol{X}^{(\mu)}\boldsymbol{Q}\boldsymbol{Q}^T, \boldsymbol{X}^{(\mu)} \right\rangle_F \\
&= \|\boldsymbol{X}^{(\mu)}\|_F^2 = \|\mathcal{X}\|_F^2 \ .
\end{aligned}$$

Taking the square root gives the desired result. □

3.3.3 The Set of Tensors with Fixed Tucker Rank

The sets of tensors with fixed or bounded Tucker rank inherit most of the useful structure of low rank matrices introduced in Section 3.1.4. In particular, it is shown by Uschmajew [68] that the set

$$\mathcal{M}_{\boldsymbol{r}}^{\mathrm{T}}(\mathbb{R}^{n_1 \times \dots \times n_d}) := \{\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d} \mid \text{T-rank}\,(\mathcal{X}) = \boldsymbol{r}\}$$

of tensors with fixed Tucker rank admits a manifold structure.[6]

Theorem 3.16 (Tucker Manifold [68, Satz 5.2])**.** *The set $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{T}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ of tensors with fixed Tucker rank forms a differentiable embedded submanifold of $\mathbb{R}^{n_1 \times \dots \times n_d}$, with dimension*

$$\dim\left(\mathcal{M}_{\boldsymbol{r}}^{\mathrm{T}}(\mathbb{R}^{n_1 \times \dots \times n_d})\right) = \prod_{\mu=1}^{d} r_\mu + \sum_{\mu=1}^{d} n_\mu r_\mu - \sum_{\mu=1}^{d} r_\mu^2 \ .$$

Analogously to the matrix case, the set $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{T}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ is not closed, but the set $\mathcal{M}_{\preceq\boldsymbol{r}}^{\mathrm{T}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ of tensors with bounded Tucker rank is closed and it is easy to see that it is the closure of $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{T}}(\mathbb{R}^{n_1 \times \dots \times n_d})$.

Proposition 3.17. *The set*

$$\mathcal{M}_{\preceq\boldsymbol{r}}^{\mathrm{T}}(\mathbb{R}^{n_1 \times \dots \times n_d}) := \{\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d} \mid \text{T-rank}\,(\mathcal{X}) \preceq \boldsymbol{r}\}$$

of tensors with bounded Tucker rank is closed.

[6]The theorem of Uschmajew [68] is even somewhat more general and does not require finite dimensions.

Proof. Matricizations are isomorphisms, therefore the sets

$$\mathcal{M}_{\mu,\leq r_\mu}(\mathbb{R}^{n_1\times\ldots\times n_d}) := \left\{\mathcal{X}\in\mathbb{R}^{n_1\times\ldots\times n_d} \;\middle|\; \operatorname{rank}\left(\boldsymbol{X}^{(\mu)}\right)\leq r_\mu\right\} ,$$

as pre-images of the closed matrix sets $\mathcal{M}_{\leq r}(\mathbb{R}^{n_\mu\times n_1\cdot\ldots\cdot n_{\mu-1}\cdot n_{\mu+1}\cdot\ldots\cdot n_d})$ from Section 3.1.4, are closed as well. Expressing the low rank decomposition

$$\mathcal{M}^{\mathrm{T}}_{\preceq \boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d}) = \bigcap_{\mu=1}^{d}\mathcal{M}_{\mu,r_\mu}(\mathbb{R}^{n_1\times\ldots\times n_d})$$

as intersection of these sets directly shows that it is closed as well. □

3.3.4 Computational Aspects

To store a tensor $\mathcal{X}\in\mathcal{M}^{\mathrm{T}}_{\preceq\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ in the Tucker representation, it is sufficient to store the core tensor $\mathcal{C}$ and the basis matrices $\boldsymbol{U}_\mu$. Setting $n=\max(n_1,\ldots,n_d)$ and $r=\max(r_1,\ldots,r_d)$, this requires on the order of $\Theta(r^d+ndr)$ memory. For $r\ll n$, this offers a major reduction compared to the general $\Theta(n^d)$ scaling, but does not prevent the exponential scaling with respect to the order d. A similar scaling can also be achieved for many common operations, if all operands are given in a low rank Tucker representation. In practice, this means that the Tucker format is mainly applicable to moderate orders, where the exponential scaling is not prohibitive.

The following list of operations is analogous to Section 3.1.5 for matrices. In the following, let $\mathcal{X},\bar{\mathcal{X}}\in\mathbb{R}^{n_1\times\ldots\times n_d}$ be two tensors of Tucker rank $\boldsymbol{r}=(r_1,\ldots,r_d)$ and $\bar{\boldsymbol{r}}=(\bar{r}_1,\ldots,\bar{r}_d)$, respectively. With exception of the normalization paragraph, we assume that $\mathcal{X}$ and $\bar{\mathcal{X}}$ are given in Tucker representations

$$\begin{aligned}\mathcal{X} &= \mathcal{C}\times_1\boldsymbol{U}_1^T\times_2\boldsymbol{U}_2^T\ldots\times_d\boldsymbol{U}_d^T\\ \bar{\mathcal{X}} &= \bar{\mathcal{C}}\times_1\bar{\boldsymbol{U}}_1^T\times_2\bar{\boldsymbol{U}}_2^T\ldots\times_d\bar{\boldsymbol{U}}_d^T ,\end{aligned}$$

with right-orthogonal $\boldsymbol{U}_\mu\in\mathbb{R}^{r_\mu\times n_\mu}$ and $\bar{\boldsymbol{U}}_\mu\in\mathbb{R}^{\bar{r}_\mu\times n_\mu}$. For all following statements on the computational complexity, define $n=\max(n_1,\ldots,n_d)$, $r=\max(r_1,\ldots,r_d)$ and $\bar{r}=\max(\bar{r}_1,\ldots,\bar{r}_d)$.

Normalization, (Re-)Orthogonalization Assume that $\mathcal{X}\in\mathbb{R}^{n_1\times\ldots\times n_d}$ is not given in a Tucker decomposition but in a low rank representation

$$\mathcal{X}=\mathcal{D}\times_1\boldsymbol{V}_1\times_2\boldsymbol{V}_2\ldots\times_d\boldsymbol{V}_d ,$$

with $\boldsymbol{V}_\mu\in\mathbb{R}^{r'_\mu\times n_\mu}$ and $\mathcal{D}\in\mathbb{R}^{r'_1\times\ldots\times r'_d}$. Here, the $\boldsymbol{V}_\mu$ are not necessarily right-orthogonal and $\boldsymbol{r}'=(r'_1,\ldots,r'_d)$ might not be the Tucker rank of $\mathcal{X}$. Transforming this to a valid Tucker representation of $\mathcal{X}$ is usually called a reorthogonalization. One possible method is to compute the RQ-decompositions $\boldsymbol{R}_\mu \boldsymbol{Q}_\mu := \boldsymbol{V}_\mu$. Then,

$$\begin{aligned}\mathcal{X} &= \mathcal{D} \times_1 \boldsymbol{V}_1 \times_2 \boldsymbol{V}_2 \ldots \times_d \boldsymbol{V}_d \\ &= \mathcal{D} \times_1 (\boldsymbol{R}_1\boldsymbol{Q}_1) \times_2 (\boldsymbol{R}_1\boldsymbol{Q}_1) \ldots \times_d (\boldsymbol{R}_1\boldsymbol{Q}_1) \\ &= \underbrace{(\mathcal{D} \times_1 \boldsymbol{R}_1 \times_2 \boldsymbol{R}_2 \ldots \times_d \boldsymbol{R}_d)}_{:=\mathcal{C}} \times_1 \boldsymbol{Q}_1 \times_2 \boldsymbol{Q}_2 \ldots \times_d \boldsymbol{Q}_d\end{aligned}$$

is a valid Tucker representation of rank $\boldsymbol{r}'$ of $\mathcal{X}$. However, the rank might still not be the actual Tucker rank. One possibility to achieve this is to perform a HOSVD on the core tensor, as described in the next paragraph. The computational cost of the reorthogonalization is composed of the QR-decompositions and the calculation of $\mathcal{C}$, and scales as $\mathcal{O}(dnr'^2 + dr'^{d+1})$.

Rank Truncation A quasi-best rank $\boldsymbol{r}' \prec \boldsymbol{r}$ approximation of $\mathcal{X}$ can be computed by performing a truncated HOSVD on the core tensor $\mathcal{C}$. Given the resulting representation

$$\mathcal{C} \approx \mathcal{D} = \mathcal{C}' \times_1 \boldsymbol{V}_1 \times_2 \boldsymbol{V}_2 \ldots \times_d \boldsymbol{V}_d$$

of the core tensor,

$$\begin{aligned}\mathcal{X}' &= \mathcal{D} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \ldots \times_d \boldsymbol{U}_d \\ &= (\mathcal{C}' \times_1 \boldsymbol{V}_1 \times_2 \boldsymbol{V}_2 \ldots \times_d \boldsymbol{V}_d) \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \ldots \times_d \boldsymbol{U}_d \\ &= \mathcal{C}' \times_1 \underbrace{(\boldsymbol{V}_1\boldsymbol{U}_1)}_{:=\boldsymbol{U}_1'} \times_2 \underbrace{(\boldsymbol{V}_2\boldsymbol{U}_2)}_{:=\boldsymbol{U}_2'} \ldots \times_d \underbrace{(\boldsymbol{V}_d\boldsymbol{U}_d)}_{:=\boldsymbol{U}_1'}\end{aligned}$$

is a rank $\boldsymbol{r}'$ Tucker representation of an approximation of $\mathcal{X}$. Applying Lemma 3.15 gives that $\mathcal{X}'$ fulfills the same quasi-optimality as a direct HOSVD, i.e.

$$\|\mathcal{X} - \mathcal{X}'\|_F \leq \sqrt{d} \cdot \min_{\text{T-rank}(\mathcal{Y}) \preceq \boldsymbol{r}'} (\|\mathcal{X} - \mathcal{Y}\|_F) \ .$$

The computational cost is made up of the HOSVD of the core tensors and the matrix-matrix products for the basis matrices and scales as $\mathcal{O}(dr^{d+1} + dnrr')$, compared to $\mathcal{O}(dn^{d+1})$ for a direct application of the HOSVD on $\mathcal{X}$.

Access Entries The entries $\mathcal{X}[i_1, \ldots, i_d]$ cannot be accessed directly in the Tucker representation. Instead, one has to compute

$$\begin{aligned}\mathcal{X}[i_1, \ldots, i_d] &= (\mathcal{C} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \ldots \times_d \boldsymbol{U}_d)[i_1, \ldots, i_d] \\ &= \sum_{p_1=1}^{r_1} \cdots \sum_{p_d=1}^{r_d} \mathcal{C}[p_1, \ldots, p_d] \boldsymbol{U}_1[p_1, i_1] \boldsymbol{U}_2[p_2, i_2] \ldots \boldsymbol{U}_d[p_d, i_d] \ .\end{aligned}$$

The computational cost scales as $\mathcal{O}(r^d)$, compared to the constant access cost for directly stored tensors.

Addition Similarly to the method for matrices, the sum $\mathcal{X} + \bar{\mathcal{X}}$ can be computed while maintaining a low rank representation at all times. For this, define

$$\boldsymbol{V}_\mu := \begin{bmatrix} \boldsymbol{U}_\mu \\ \bar{\boldsymbol{U}}_\mu \end{bmatrix} \in \mathbb{R}^{(r_\mu + \bar{r}_\mu) \times n_\mu}$$

and

$$\mathcal{D}[i_1, \ldots, i_d] := \begin{cases} \mathcal{C}[i_1, \ldots, i_d] & i_\mu \leq r_\mu \forall \mu \\ \bar{\mathcal{C}}[i_1 - r_1, \ldots, i_d - r_d] & i_\mu > r_\mu \forall \mu \\ 0 & \text{else} \end{cases} \qquad \in \mathbb{R}^{r_1 + \bar{r}_1 \times \ldots \times r_d + \bar{r}_d} .$$

Then,

$$\begin{aligned} &(\mathcal{D} \times_1 \boldsymbol{V}_1 \ldots \times_d \boldsymbol{V}_d)\,[i_1, \ldots, i_d] \\ &= \sum_{p_1=1}^{r_1+\bar{r}_1} \cdots \sum_{p_d=1}^{r_d+\bar{r}_d} \mathcal{D}[p_1, \ldots, p_d] \boldsymbol{V}_1[p_1, i_1] \ldots \boldsymbol{V}_d[p_d, i_d] \\ &= \sum_{p_1=1}^{r_1} \cdots \sum_{p_d=1}^{r_d} \mathcal{C}[p_1, \ldots, p_d] \boldsymbol{U}_1[p_1, i_1] \ldots \boldsymbol{U}_d[p_d, i_d] \\ &\quad + \sum_{p_1=r_1+1}^{r_1+\bar{r}_1} \cdots \sum_{p_d=r_d+1}^{r_d+\bar{r}_d} \bar{\mathcal{C}}[p_1, \ldots, p_d] \bar{\boldsymbol{U}}_1[p_1, i_1] \ldots \bar{\boldsymbol{U}}_d[p_d, i_d] \\ &= \left(\mathcal{X} + \bar{\mathcal{X}}\right)[i_1, \ldots, i_d] \end{aligned}$$

is a rank $r + \bar{r}$ representation of the sum. Note that $r + \bar{r}$ might not be the Tucker rank and that the matrices $\boldsymbol{V}_\mu$ are not necessarily right-orthogonal, possibly requiring a further reorthogonalization and rank reduction. The computational cost of this addition itself scales as $\mathcal{O}((r + \bar{r})^d + dn(r + \bar{r}))$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

Hadamard Product The entrywise product of $\mathcal{X}$ and $\bar{\mathcal{X}}$ can be computed while maintaining a low rank representation. For this, define $V_\mu \in \mathbb{R}^{r_\mu \bar{r}_\mu \times n_\mu}$ and $\mathcal{D} \in \mathbb{R}^{r_1 \bar{r}_1 \times \ldots \times r_d \bar{r}_d}$ as

$$\boldsymbol{V}_\mu[p \bar{r}_\mu + q_\mu, i_\mu] := \boldsymbol{U}_\mu[p_\mu, i_\mu] \bar{\boldsymbol{U}}_\mu[q_\mu, i_\mu] \qquad 1 \leq p_\mu \leq r_\mu,\ 1 \leq q_\mu \leq \bar{r}_\mu$$

and

$$\mathcal{D}[p_1 \bar{r}_1 + q_1, \ldots, p_d \bar{r}_d + q_d] := \mathcal{C}[p_1, \ldots, p_d]\, \bar{\mathcal{C}}[q_1, \ldots, q_d] \quad 1 \leq p_\mu \leq r_\mu,\ 1 \leq q_\mu \leq \bar{r}_\mu .$$

Then,

$$
\begin{aligned}
&\left(\mathcal{D} \times_1 \boldsymbol{V}_1 \dots \times_d \boldsymbol{V}_d\right)[i_1, \dots i_d] \\
&= \sum_{p_1=1}^{r_1\bar{r}_1} \dots \sum_{p_d=1}^{r_d\bar{r}_d} \mathcal{D}[p_1, \dots, p_d] \boldsymbol{V}_1[p_1, i_1] \dots \boldsymbol{V}_d[p_d, i_d] \\
&= \sum_{p_1=1}^{r_1} \dots \sum_{p_d=1}^{r_d} \sum_{q_1=1}^{\bar{r}_1} \dots \sum_{q_d=1}^{\bar{r}_d} \mathcal{D}[p_1\bar{r}_1 + q_1, \dots, p_d\bar{r}_d + q_d] \boldsymbol{V}_1[p_1\bar{r}_1 + q_1, i_1] \dots \boldsymbol{V}_d[p_d\bar{r}_d + q_d, i_d] \\
&= \sum_{p_1,\dots,p_d} \mathcal{C}[p_1, \dots, p_d] \boldsymbol{U}_1[p_1, i_1] \dots \boldsymbol{U}_d[p_d, i_d] \cdot \sum_{q_1,\dots,q_d} \bar{\mathcal{C}}[q_1, \dots, q_d] \bar{\boldsymbol{U}}_1[q_1, i_1] \dots \bar{\boldsymbol{U}}_d[q_d, i_d] \\
&= \mathcal{X}[i_1, \dots, i_d] \cdot \bar{\mathcal{X}}[i_1, \dots, i_d]
\end{aligned}
$$

is a rank $(r_1\bar{r}_1, \dots, r_d\bar{r}_d)$ representation of the entrywise product. The matrices $\boldsymbol{V}_\mu$ are not necessarily right-orthogonal and the rank might not be optimal, advising an additional reorthogonalization and rank reduction step. The computational complexity scales as $\mathcal{O}(dnr\bar{r} + r^d\bar{r}^d)$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

Frobenius Inner Product The Frobenius scalar product of $\mathcal{X}$ and $\bar{\mathcal{X}}$ can be calculated via

$$
\begin{aligned}
&\left\langle \mathcal{X},\, \bar{\mathcal{X}} \right\rangle_F \\
&= \left\langle \mathcal{C} \times_1 \boldsymbol{U}_1 \dots \times_d \boldsymbol{U}_d,\ \bar{\mathcal{C}} \times_1 \bar{\boldsymbol{U}}_1 \dots \times_d \bar{\boldsymbol{U}}_d \right\rangle_F \\
&= \sum_{i_1,\dots,i_d} \sum_{p_1,\dots,p_d} \sum_{q_1,\dots,q_d} \mathcal{C}[p_1, \dots, p_d] \boldsymbol{U}_1[p_1, i_1] \dots \boldsymbol{U}_d[p_d, i_d] \bar{\mathcal{C}}[q_1, \dots, q_d] \bar{\boldsymbol{U}}_1[q_1, i_1] \dots \bar{\boldsymbol{U}}_d[q_d, i_d] \\
&= \sum_{p_1,\dots,p_d} \mathcal{C}[p_1, \dots, p_d] \left(\sum_{q_1,\dots,q_d} \bar{\mathcal{C}}[q_1, \dots, q_d] \left(\boldsymbol{U}_1\bar{\boldsymbol{U}}_1^T\right)[p_1, q_1] \dots \left(\boldsymbol{U}_d\bar{\boldsymbol{U}}_d^T\right)[p_d, q_d] \right) .
\end{aligned}
$$

This way, the computational costs scale as $(dnr\bar{r}+dr\bar{r}^d+r^d)$, compared to $\mathcal{O}(n^d)$ for directly stored tensors. Using the symmetry, this can be improved to $\mathcal{O}(dr\bar{r}(n + \min(r^{d-1}, \bar{r}^{d-1})) + \max(r^d, \bar{r}^d))$.

Frobenius Norm From Lemma 3.15, it follows that the norm can be calculated as

$$
\|\mathcal{X}\|_F = \|\mathcal{C}\|_F .
$$

The computational complexity scales as $\mathcal{O}(r^d)$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

μ-th Mode Product Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n_\mu \times m}$, the μ-th mode product $\mathcal{X} \times_\mu \boldsymbol{A}$ can be computed as

$$\begin{aligned}\mathcal{X} \times_\mu \boldsymbol{A} &= (\mathcal{C} \times_1 \boldsymbol{U}_1 \ldots \times_d \boldsymbol{U}_d) \times_\mu \boldsymbol{A} \\ &= \mathcal{C} \times_1 \boldsymbol{U}_1 \ldots \times_\mu \underbrace{(\boldsymbol{U}_\mu \boldsymbol{A})}_{:=\boldsymbol{B} \in \mathbb{R}^{r_\mu \times m}} \ldots \times_d \boldsymbol{U}_d \,.\end{aligned}$$

This is a low rank representation of $\mathcal{X} \times_\mu \boldsymbol{A}$ with the same rank $\boldsymbol{r}$. To obtain a valid Tucker representation, calculate the RQ-decomposition $\boldsymbol{RQ} = \boldsymbol{B}$ and $\mathcal{C}' = \mathcal{C} \times_\mu \boldsymbol{R}$. Then,

$$\mathcal{X} \times_\mu \boldsymbol{A} = \mathcal{C}' \times_1 \boldsymbol{U}_1 \ldots \times_\mu \boldsymbol{Q} \ldots \times_d \boldsymbol{U}_d$$

is a valid Tucker representation of the same rank. Note, however, that this rank might not be the Tucker rank. The scaling of computational complexity amounts to $\mathcal{O}(mnr + mr^2 + r^{d+1})$, compared to $\mathcal{O}(mn^d)$ for directly stored tensors.

Other Contractions General contractions can also be computed while remaining in a low rank representation at all times, if both operands are given as a Tucker representation. The following shows this for the contraction of a single mode pair. The contraction of multiple mode pairs can be shown analogously.

Let $\mathcal{A} \in \mathbb{R}^{m_1 \times \ldots \times m_e}$ be another tensor of order e given in a Tucker representation $\mathcal{A} = \mathcal{D} \times_1 \boldsymbol{V}_1 \ldots \times_d \boldsymbol{V}_d$ with rank $(r'_1, \ldots, r'_e)$. Assuming $m_\mu = n_\nu$ holds, the contraction $\mathcal{A} *_{\mu,\nu} \mathcal{X}$ of the μ-th mode of $\mathcal{A}$ with the ν-th mode of $\mathcal{X}$ can be calculated as

$$\begin{aligned}&(\mathcal{A} *_{\mu,\nu} \mathcal{X})\,[j_1, \ldots, j_{\mu-1}, j_{\mu+1}, \ldots, j_e, i_1, \ldots, i_{\nu-1}, i_{\nu+1}, \ldots, i_d] \\ &= \sum_{k=1}^{m_\mu} \sum_{p_1,\ldots,p_e} \mathcal{D}[p_1, \ldots, p_e] \boldsymbol{V}_1[p_1, j_1] \ldots \boldsymbol{V}_\mu[p_\mu, k] \ldots \boldsymbol{V}_e[p_e, j_e] \\ &\quad \cdot \sum_{q_1,\ldots,q_e} \mathcal{C}[q_1, \ldots, q_d] \boldsymbol{U}_1[q_1, i_1] \ldots \boldsymbol{U}_\nu[q_\nu, k] \ldots \boldsymbol{U}_d[q_d, i_d] \\ &= \sum_{\substack{p_1,\ldots,p_{\mu-1},p_{\mu+1},\ldots,p_e, \\ q_1,\ldots,q_{\nu-1},q_{\nu+1},\ldots,q_d}} \underbrace{\left(\sum_{k,p_\mu,q_\nu} \mathcal{D}[p_1, \ldots, p_e] C[q_1, \ldots, q_d] \boldsymbol{V}_\mu[p_\mu, k] \boldsymbol{U}_\nu[q_\nu, k] \right)}_{:=\mathcal{K}[p_1,\ldots,p_{\mu-1},p_{\mu+1},\ldots,p_e,q_1,\ldots,q_{\nu-1},q_{\nu+1},\ldots,q_d]} \\ &\quad \cdot \boldsymbol{V}_1[p_1, j_1] \ldots \boldsymbol{V}_{\mu-1}[p_{\mu-1}, j_{\mu-1}] \boldsymbol{V}_{\mu+1}[p_{\mu+1}, j_{\mu+1}] \ldots \boldsymbol{V}_e[p_e, j_e] \\ &\quad \cdot \boldsymbol{U}_1[q_1, i_1] \ldots \boldsymbol{U}_{\nu-1}[q_{\nu-1}, i_{\nu-1}] \boldsymbol{U}_{\nu+1}[q_{\nu+1}, i_{\nu-+}] \ldots \boldsymbol{U}_d[q_d, i_d] \,.\end{aligned}$$

The final term is a valid Tucker representation of $\mathcal{A} *_{\mu,\nu} \mathcal{X}$ of representation rank $(r'_1, \ldots, r'_{\mu-1}, r'_{\mu+1}, \ldots, r'_e, r_1, \ldots, r_{\nu-1}, r_{\nu+1}, \ldots, r_d)$. Calculating the matrix product $\boldsymbol{V}_\mu^T \boldsymbol{U}_\nu$ first, the computational complexity scales as $\mathcal{O}(m_\mu r'_\mu r_\nu + \min(r'_\mu, r_\nu) r'^{e-1} r^{d-1})$, compared to $\mathcal{O}(\min(m, n) m^{e-1} n^{d-1})$ for directly stored tensors. As with most operations, this rank might not be minimal and can be reduced by a subsequent rank reduction step.

3.4 Tensor Train Decomposition

The tensor train (TT) decomposition follows a subspace based approach similar to the Tucker format. In this way, the TT format retains most of the advantages of the matrix SVD and the Tucker format, namely a generalized higher order SVD, a closed set of low rank tensors and a manifold structure on the set of tensors with fixed rank. At the same time, the computational complexity of most common operations scales only *linearly* in the order if all operands are given in TT representation. This means that to some extent, the tensor train format unifies the advantages of both the canonical and the Tucker format.

The history of the tensor train format is shaped by several independent re-inventions of the format. In particular, it was already known for almost half a century under the name "matrix product states" (MPS) in quantum physics. In this context, MPS are especially useful to describe many-particle quantum states with low entanglement. They are at the very heart of the extremely successful DMRG algorithm, originally introduced by White [85] in 1992. A comprehensive introduction to MPS from a physical point of view is given by Perez-Garcia et al. [21], in combination with the DMRG algorithm by Schollwöck [22]. Despite the tremendous success of MPS in physics, the concept was largely unknown to most of the mathematical community. To this community, the concept of MPS was introduced by Oseledets [20], who developed it as a means of tensor decomposition. In this work, the name tensor train format was coined. These reinventions resulted in different sets of names and notations. This thesis predominantly uses the conventions of the mathematical community.

3.4.1 Definition

Similar to the Tucker decomposition, the idea of the tensor train (TT) decomposition is to find minimal subspaces, such that a given tensor can be represented in terms of these spaces. But in contrast to the Tucker format, these subspaces are not chosen as corresponding to the single modes. Instead, a hierarchy of nested subspaces with increasing order is constructed, such that the final subspace contains the given tensor. More precisely, the construction is done by finding (minimal) subspaces $U_1, \ldots, U_{d-1}$, such that for a given $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$,

$$\begin{aligned}
U_1 &\subseteq \mathbb{R}^{n_1} && \text{with } \mathcal{X} \in U_1 \otimes \mathbb{R}^{n_2 \times \ldots \times n_d} \\
U_2 &\subseteq U_1 \otimes \mathbb{R}^{n_2} \subseteq \mathbb{R}^{n_1 \times n_2} && \text{with } \mathcal{X} \in U_2 \otimes \mathbb{R}^{n_3 \times \ldots \times n_d} \\
U_3 &\subseteq U_2 \otimes \mathbb{R}^{n_3} \subseteq \mathbb{R}^{n_1 \times n_2 \times n_3} && \text{with } \mathcal{X} \in U_3 \otimes \mathbb{R}^{n_4 \times \ldots \times n_d} \\
&\vdots \\
U_{d-1} &\subseteq \mathbb{R}^{n_{d-1}} \otimes U_{d-2} \subseteq \mathbb{R}^{n_1 \times \ldots \times n_{d-1}} && \text{with } \mathcal{X} \in U_{d-1} \otimes \mathbb{R}^{n_d} \; .
\end{aligned} \tag{3.8}$$

Let r_μ denote the dimension of U_μ and let $\{\mathcal{V}_{\mu,1} \ldots \mathcal{V}_{\mu,r_\mu}\}$ be orthogonal bases for U_μ. For $\mu > 1$, the nested construction $U_\mu \subseteq U_{\mu-1} \otimes \mathbb{R}^{n_\mu}$ ensures that there exist vectors

$\boldsymbol{u}_{\mu,i,k} \in \mathbb{R}^{n_\mu}$, such that

$$\mathcal{V}_{\mu,k} = \sum_{i=1}^{r_{\mu-1}} \mathcal{V}_{\mu-1,i} \otimes \boldsymbol{u}_{\mu,i,k} \ . \tag{3.9}$$

Writing the orthogonal basis as a tensor

$$\mathcal{W}_\mu[j_1,\ldots,j_\mu,k] := \mathcal{V}_{\mu,k}[j_1,\ldots,j_\mu]$$

and defining

$$\mathcal{U}_\mu[i,j,k] := \boldsymbol{u}_{\mu,i,k}[j] \ ,$$

Equation (3.9) can be rewritten as

$$\begin{aligned}
\mathcal{W}_\mu[i_1,\ldots,i_\mu,k] &= \sum_{i=1}^{r_{\mu-1}} \mathcal{V}_{\mu-1,i}[j_1,\ldots,j_{\mu-1}]\boldsymbol{u}_{\mu,i,k}[j_\mu] \\
&= \sum_{i=1}^{r_{\mu-1}} \mathcal{W}_{\mu-1}[j_1,\ldots,j_{\mu-1},i]\boldsymbol{u}_{\mu,i,k}[j_\mu] \\
&= \sum_{i=1}^{r_{\mu-1}} \mathcal{W}_{\mu-1}[j_1,\ldots,j_{\mu-1},i]\mathcal{U}_\mu[i,j_\mu,k] \\
&= \left(\mathcal{W}_{\mu-1} \circ \mathcal{U}_\mu\right)[j_1,\ldots,j_\mu,k] \ .
\end{aligned} \tag{3.10}$$

Since $\mathcal{X} \in U_{d-1} \otimes \mathbb{R}^{n_d}$, there also exist vectors $\boldsymbol{u}_{d,i} \in \mathbb{R}^{n_d}$, such that

$$\mathcal{X} = \sum_{i=1}^{r_{d-1}} \mathcal{V}_{d-1,i} \otimes \boldsymbol{u}_{d,i} \ .$$

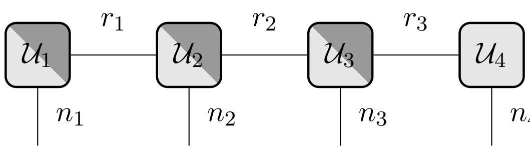
Similar to the above, defining

$$\mathcal{U}_d[i,j] := \boldsymbol{u}_{d,i}[j] \ ,$$

this can be rewritten as $\mathcal{X} = \mathcal{W}_{d-1} \circ \mathcal{U}_d$. Applying Equation (3.10) recursively gives

$$\begin{aligned}
\mathcal{X} &= \mathcal{W}_{d-1} \circ \mathcal{U}_d \\
&= \left(\mathcal{W}_{d-2} \circ \mathcal{U}_{d-1}\right) \circ \mathcal{U}_d \\
&= \left(\mathcal{W}_{d-3} \circ \mathcal{U}_{d-2}\right) \circ \mathcal{U}_{d-1} \circ \mathcal{U}_d \\
&\vdots \\
&= \mathcal{W}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_{d-1} \circ \mathcal{U}_d \\
&= \mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_{d-1} \circ \mathcal{U}_d \ ,
\end{aligned} \tag{3.11}$$

where we defined $\mathcal{U}_1 := \mathcal{W}_1$. Note that the tensors $\mathcal{U}_1,\ldots,\mathcal{U}_{d-1}$ completely define the corresponding subspaces $U_1,\ldots U_{d-1}$. Equation (3.11) can be interpreted as a tensor network and as such be visualized in the diagrammatic notation. The following shows this by example for an order four tensor.



Here, the diagrammatic notation already indicates that for $1 \leq \mu < d$, the $\mathcal{U}_\mu$ are left-orthogonal. In particular, it holds that

$$\boldsymbol{U}_1^{\langle 1 \rangle^T} \boldsymbol{U}_1^{\langle 1 \rangle} = \boldsymbol{I} \in \mathbb{R}^{r_1 \times r_1}$$

and for $1 < \mu < d$,

$$\boldsymbol{U}_\mu^{\langle 2 \rangle^T} \boldsymbol{U}_\mu^{\langle 2 \rangle} = \boldsymbol{I} \in \mathbb{R}^{r_\mu \times r_\mu} \ .$$

For $\mathcal{U}_1 = \mathcal{W}_1$, this is obvious from the construction as a matrix containing orthonormal basis vectors. For $1 < \mu < d$, note that

$$\begin{aligned}
\left(\boldsymbol{W}_\mu^{\langle \mu \rangle^T} \boldsymbol{W}_\mu^{\langle \mu \rangle}\right)[k_1, k_2] &= \sum_{i_1,\dots,i_\mu} \mathcal{W}_\mu[i_1, \dots, i_\mu, k_1] \mathcal{W}_\mu[i_1, \dots, i_\mu, k_2] \\
&= \sum_{i_1,\dots,i_\mu} \mathcal{V}_{\mu,k_1}[i_1, \dots, i_\mu] \mathcal{V}_{\mu,k_2}[i_1, \dots, i_\mu] \\
&= \langle \mathcal{V}_{\mu,k_1},\ \mathcal{V}_{\mu,k_2} \rangle = \delta_{k_1,k_2} \ ,
\end{aligned}$$

because the $\mathcal{V}_{\mu,k}$ form an orthonormal basis by construction. It follows that

$$\begin{aligned}
\boldsymbol{I}[k_1, k_2] &= \left(\boldsymbol{W}_\mu^{\langle \mu \rangle^T} \boldsymbol{W}_\mu^{\langle \mu \rangle}\right)[k_1, k_2] \\
&= \sum_{i_1,\dots,i_\mu} \left(\mathcal{W}_{\mu-1} \circ \mathcal{U}\right)[i_1, \dots, i_\mu, k_1] \left(\mathcal{W}_{\mu-1} \circ \mathcal{U}\right)[i_1, \dots, i_\mu, k_2] \\
&= \sum_{i_1,\dots,i_\mu} \sum_{j_1,j_2} \mathcal{W}_{\mu-1}[i_1, \dots, i_{\mu-1}, j_1] \mathcal{U}[j_1, i_\mu, k_1] \mathcal{W}_{\mu-1}[i_1, \dots, i_{\mu-1}, j_2] \mathcal{U}[j_1, i_\mu, k_1] \\
&= \sum_{i_\mu} \sum_{j_1,j_2} \boldsymbol{I}[j_1, j_2] \mathcal{U}[j_1, i_\mu, k_1] \mathcal{U}[j_1, i_\mu, k_1] \\
&= \left(\boldsymbol{U}_\mu^{\langle 2 \rangle^T} \boldsymbol{U}_\mu^{\langle 2 \rangle}\right)[k_1, k_2] \ ,
\end{aligned}$$

which proves the assumption.

Note that the choice of the subspace hierarchy in (3.8) was somewhat arbitrary. In fact, for any $1 \leq \nu \leq d$, one could choose subspaces $U_1, \dots U_{\nu-1}, U_{\nu+1} \dots, U_d$, such that for a given $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$

$$\begin{aligned}
U_1 &\subseteq \mathbb{R}^{n_1} && \text{with } \mathcal{X} \in U_1 \otimes \mathbb{R}^{n_2 \times \dots \times n_d} \\
U_2 &\subseteq U_1 \otimes \mathbb{R}^{n_2} \subseteq \mathbb{R}^{n_1 \times n_2} && \text{with } \mathcal{X} \in U_2 \otimes \mathbb{R}^{n_3 \times \dots \times n_d} \\
&\vdots \\
U_{\nu-1} &\subseteq U_{\nu-2} \otimes \mathbb{R}^{n_{\nu-1}} \subseteq \mathbb{R}^{n_1 \times \dots \times n_{\nu-1}} && \text{with } \mathcal{X} \in U_{\nu-1} \otimes \mathbb{R}^{n_\nu \times \dots \times n_d} \\
U_{\nu+1} &\subseteq \mathbb{R}^{n_{\nu+1}} \otimes U_{\nu+2} \subseteq \mathbb{R}^{n_{\nu+1} \times \dots \times n_d} && \text{with } \mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_\nu} \otimes U_{\nu+1} \\
&\vdots \\
U_{d-1} &\subseteq \mathbb{R}^{n_{d-1}} \otimes U_d \subseteq \mathbb{R}^{n_{d-1} \times n_d} && \text{with } \mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_{d-2}} \otimes U_{d-1} \\
U_d &\subseteq \mathbb{R}^{n_d} && \text{with } \mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_{d-1}} \otimes U_d
\end{aligned}$$

and consequently $\mathcal{X} \in U_{\nu-1} \otimes \mathbb{R}^{n_\nu} \otimes U_{\nu+1}$. Repeating the same construction as above yields left-orthogonal $\mathcal{U}_1, \ldots \mathcal{U}_{\nu-1}$, right-orthogonal $\mathcal{U}_{\nu+1}, \ldots \mathcal{U}_d$ and $\mathcal{U}_\nu$ such that

$$\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d$$

holds. As will be shown in Section 3.4.4, the position of the non-orthogonal tensor can be changed *without* altering the subspace dimensions, which means that the different subspace hierarchies are equivalent for most purposes. In particular, this allows the following formal definition of the tensor train representation, which is based on the above subspace construction.

Definition 3.18 (Tensor Train Format)**.** Let $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ be a tensor of order d. A factorization

$$\mathcal{X} = \mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_{d-1} \circ \mathcal{U}_d \tag{3.12}$$

of $\mathcal{X}$ into *component tensors* $\mathcal{U}_1 \in \mathbb{R}^{n_1 \times r_1}$, $\mathcal{U}_\mu \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}, \mu = 2, \ldots, d-1$ and $\mathcal{U}_d \in \mathbb{R}^{r_{d-1}, n_d}$ is called a tensor train (TT) representation of $\mathcal{X}$. The tuple of the dimensions $(r_1, \ldots, r_{d-1})$ of the component tensors is called the representation rank and is associated with the specific representation. In contrast, the tensor train rank (TT rank) is defined as the minimal rank tuple[7] $\boldsymbol{r} = (r_1, \ldots, r_d)$, such that there exists a TT representation of $\mathcal{X}$ with representation rank equal to $\boldsymbol{r}$. The representation is said to be orthogonalized if for some ν, the component tensors $\mathcal{U}_1, \ldots, \mathcal{U}_{\nu-1}$ are left-orthogonal and $\mathcal{U}_{\nu+1}, \ldots, \mathcal{U}_d$ are right-orthogonal. The index ν of the (possibly) non-orthogonal component tensor is called the core position.

An alternative way to interpret the component tensors $\mathcal{U}_\mu$ is as sets of vectors and matrices defined as

$$\begin{aligned} \boldsymbol{U}_{1,i_1}[j_1] &:= \mathcal{U}_1[i_1] \\ \boldsymbol{U}_{\mu,i_\mu}[j_{\mu-1}, j_\mu] &:= \mathcal{U}_\mu[j_{\mu-1}, i_\mu, j_\mu] \qquad\qquad \mu = 2, \ldots, d-1 \\ \boldsymbol{U}_{d,i_d}[j_{d-1}] &:= \mathcal{U}_d[j_{d-1}, i_d] \ . \end{aligned}$$

Thereby, (3.12) can be given as

$$\mathcal{X}[i_1, \ldots, i_d] = \boldsymbol{U}_{1,i_1} \, \boldsymbol{U}_{2,i_2} \ldots \boldsymbol{U}_{d-1,i_{d-1}} \, \boldsymbol{U}_{d,i_d} \ , \tag{3.13}$$

where the juxtaposition denotes the regular matrix product. This notation is especially popular in the theoretical physics community, where the tensor train format is known as *matrix product states.*

From the derivation of the TT format at the beginning of this section, it should be clear that there always exists a TT representation of any given tensor. For example, taking each subspace $U_\mu = \mathbb{R}^{n_1 \times \cdots \times n_\mu}$ equal to the complete space always works. However, it is *a priori* not clear whether there always is a TT representation with a minimal rank in the

[7]In the sense of Definition 3.12

sense of Definition 3.12. As with the Tucker format, the proof can be established using a generalized singular value decomposition, which is done in the following section. For this, we also require the following lemma.

Lemma 3.19. *Given left-orthogonal tensors $\mathcal{V} \in \mathbb{R}^{n_1 \times \dots \times n_{\mu-1} \times r_{\mu-1}}$ and $\mathcal{W} \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}$, the contraction $\mathcal{X} = \mathcal{V} \circ \mathcal{W}$ is also left-orthogonal. Analogously, given right-orthogonal tensors $\mathcal{W} \in \mathbb{R}^{r_{\mu-1} \times n_{\mu-1} \times r_\mu}$ and $\mathcal{V} \in \mathbb{R}^{r_\mu \times n_\mu \times \dots \times n_d}$, the contraction $\mathcal{X} = \mathcal{W} \circ \mathcal{V}$ is also right-orthogonal.*

Proof. We prove the first statement, the second follows analogously. We have

$$
\begin{aligned}
&\left(\boldsymbol{X}^{<\mu>^T} \boldsymbol{X}^{<\mu>}\right)[k, k'] \\
&= \sum_{i_1,\dots,i_{\mu-1}} (\mathcal{V} \circ \mathcal{W})[i_1, \dots, i_\mu, k]\, (\mathcal{V} \circ \mathcal{W})[i_1, \dots, i_{\mu-1}, k'] \\
&= \sum_{i_1,\dots,i_\mu} \sum_{j_1} \mathcal{V}[i_1, \dots, i_{\mu-1}, j_1]\; \mathcal{W}[j_1, i_\mu, k]\; \sum_{j_2} \mathcal{V}[i_1, \dots, i_{\mu-1}, j_2]\; \mathcal{W}[j_2, i_\mu, k'] \\
&= \sum_{j_1} \sum_{j_2} \underbrace{\sum_{i_1,\dots,i_{\mu-1}} \mathcal{V}[i_1, \dots, i_{\mu-1}, j_1] \mathcal{V}[i_1, \dots, i_{\mu-1}, j_2]}_{=\left(\boldsymbol{V}^{\langle \mu-1 \rangle^T} \boldsymbol{V}^{\langle \mu-1 \rangle}\right)[j_1, j_2] = \boldsymbol{I}[j_1, j_2]} \sum_{i_\mu} \mathcal{W}[j_1, i_\mu, k]\; \mathcal{W}[j_2, i_\mu, k'] \\
&= \sum_{j} \sum_{i_\mu} \mathcal{W}[j_1, i_\mu, k]\; \mathcal{W}[j_2, i_\mu, k'] \\
&= \left(\boldsymbol{W}^{\langle 2 \rangle^T} \boldsymbol{W}^{\langle 2 \rangle}\right)[k, k'] \\
&= \boldsymbol{I}[k, k']\ .
\end{aligned}
$$

□

3.4.2 Calculation of a Decomposition

Due to the subspace based construction, the Tensor Train format shares a major feature with the Tucker format: There is a generalized singular value decomposition, called TT-SVD, which computes a minimal TT representation of any given tensor. Algorithm 3 gives the formal TT-SVD procedure and Figure 3.2 depicts the steps of the algorithm in the diagrammatic notation for an order four tensor. The following theorem and corollary show that the decomposition obtained by the TT-SVD is of minimal rank and can also be used as a constructive proof, showing that the TT rank is indeed well defined.

Theorem 3.20. *For every given tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, the TT-SVD obtains a valid right-orthogonal tensor train representation of $\mathcal{X}$. The representation rank is related to the ranks of matricizations of $\mathcal{X}$ via*

$$
r_\mu = \operatorname{rank}\left(\boldsymbol{X}^{\langle \mu \rangle}\right)\ . \tag{3.14}
$$

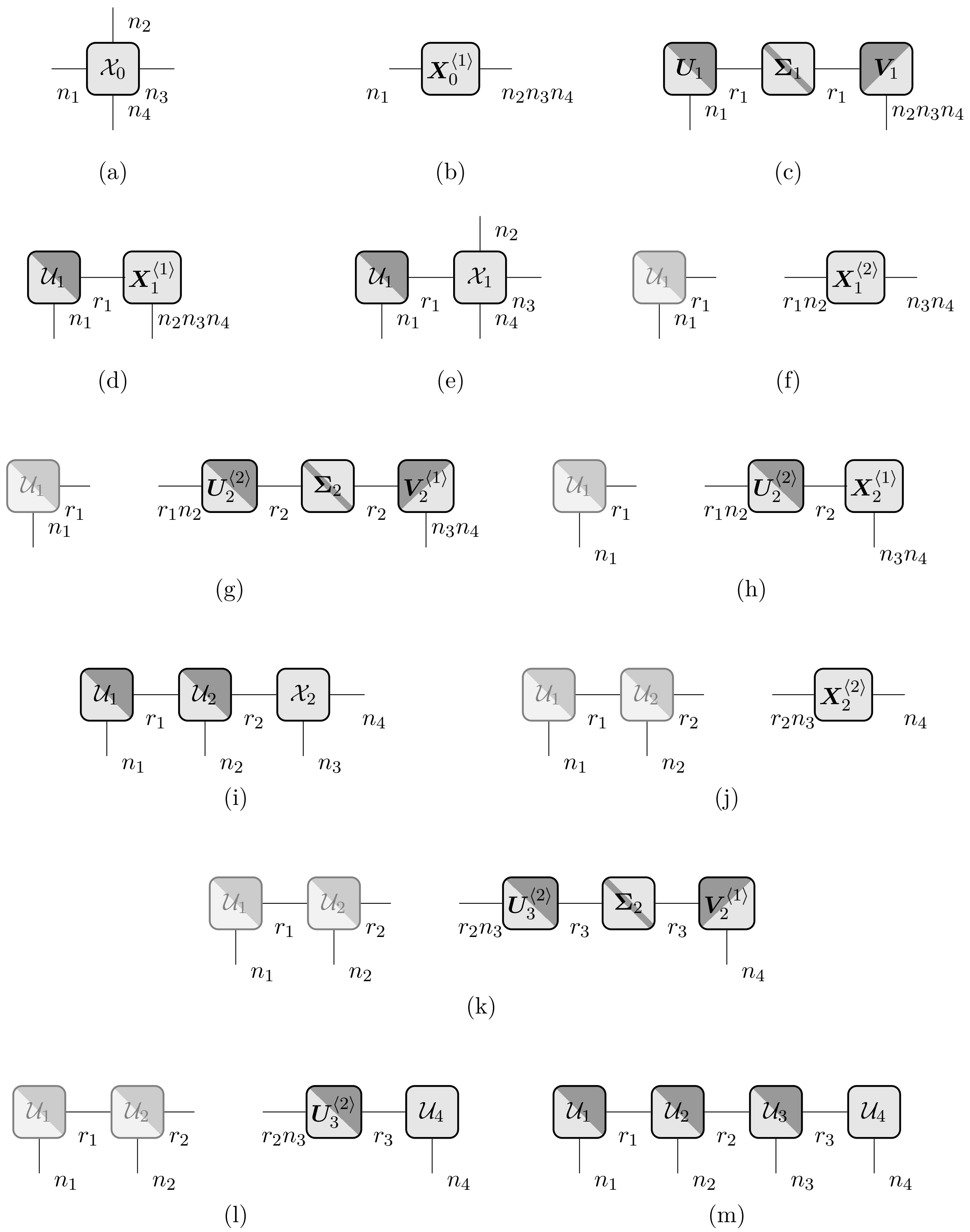


Figure 3.2: Step by step depiction of the TT-SVD by example for an order 4 tensor $\mathcal{X}_0 \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$ (a). First, a matricization combining all but one mode is performed (b) and an SVD of this matricization is calculated (c). The two rightmost matrices are contracted (d) and the result is dematricized (e). In this way, one open index is detached. The remaining tensor $\mathcal{X}_1$ is again matricized where the first dimension is formed by the internal and the next open mode (f). Again, an SVD of this matrix is calculated (g) to detach another open mode (h)-(i). This process is continued until all open modes are separated (j)-(m).

Algorithm 3: Tensor Train Singular Value Decomposition (TT-SVD)

```
Input  : Target $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$
Output : Component tensors $\mathcal{U}_1 \in \mathbb{R}^{n_1 \times r_1}, \mathcal{U}_2 \in \mathbb{R}^{r_1 \times n_2 \times r_2}, \ldots, \mathcal{U}_d \in \mathbb{R}^{r_{d-1} \times n_d}$
1 Set $\mathcal{X}_1 = \mathcal{X}$
2 Calculate SVD $\boldsymbol{U}_1^{\langle 1 \rangle} \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^{\langle 1 \rangle} = \boldsymbol{X}_1^{\langle 1 \rangle}$,
  where $\mathcal{U}_1 \in \mathbb{R}^{n_1 \times r_1}, \boldsymbol{\Sigma}_1 \in \mathbb{R}^{r_1 \times r_1}, \mathcal{V}_1 \in \mathbb{R}^{r_1 \times n_2 \times \ldots \times n_d}$
3 Set $\mathcal{X}_2 = \Sigma_1 \circ \mathcal{V}_1$
4 for $\mu = 2, \ldots, d-1$ do
5     Calculate SVD $\boldsymbol{U}_\mu^{\langle 2 \rangle} \boldsymbol{\Sigma}_\mu \boldsymbol{V}_\mu^{\langle 1 \rangle} = \boldsymbol{X}_\mu^{\langle 2 \rangle}$,
      where $\mathcal{U}_\mu \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}, \boldsymbol{\Sigma}_\mu \in \mathbb{R}^{r_\mu \times r_\mu}, \mathcal{V}_\mu \in \mathbb{R}^{r_\mu \times n_{\mu+1} \times \ldots \times n_d}$
6     Set $\mathcal{X}_{\mu+1} = \boldsymbol{\Sigma}_\mu \circ \mathcal{V}_\mu = \mathcal{U}_\mu *_{(1,2),(1,2)} \mathcal{X}_\mu$
7 Set $\mathcal{U}_d = \mathcal{X}_d$
```

Proof. The intuition of the TT-SVD is to perform a matrix SVD in every step in order to detach one mode from the tensor. Figure 3.2 shows the TT-SVD step-by-step for an order four tensor and is frequently referred to in the following. The procedure starts by calculating a SVD of the matricization of $\mathcal{X} = \mathcal{X}_0$, where all modes but the first one are combined (Fig. 3.2 (a)-(c)), i.e.

$$\boldsymbol{U}_1^{\langle 1 \rangle} \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^{\langle d-1 \rangle^T} := \mathrm{SVD}\left(\boldsymbol{X}_0^{\langle 1 \rangle}\right) \qquad \mathcal{U}_1 \in \mathbb{R}^{n_1 \times r_1}, \boldsymbol{\Sigma}_1 \in \mathbb{R}^{r_1 \times r_1}, \mathcal{V}_1 \in \mathbb{R}^{n_2 \times \ldots \times n_d \times r_1} \ .$$

A fundamental property of the compact SVD is that the dimension r_1 is equal to the rank of $\boldsymbol{X}_0^{\langle 1 \rangle} = \boldsymbol{X}^{\langle 1 \rangle}$, i.e. (3.14) holds for $\mu = 1$. The next step is to set

$$\boldsymbol{X}_1^{\langle 1 \rangle} := \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^{\langle d-1 \rangle^T} \qquad \mathcal{X}_1 \in \mathbb{R}^{r_1 \times n_2 \times \ldots \times n_d} \ .$$

Note that the following holds:

$$\mathcal{U}_1 \circ \mathcal{X}_1 = \mathrm{Mat}^{-1}\left(\boldsymbol{U}_1^{\langle 1 \rangle} \boldsymbol{X}_1^{\langle 1 \rangle}\right) = \mathrm{Mat}^{-1}\left(\boldsymbol{U}_1^{\langle 1 \rangle} \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^{\langle d-1 \rangle^T}\right) = \mathcal{X}$$

In the next step, a matricization of the newly acquired tensor $\mathcal{X}_1$ is performed. The first mode of the matricization is formed by the first two modes of $\mathcal{X}_1$, corresponding to the new dimension introduced by the prior SVD and the second original dimension. The second mode of the matricization is formed by all remaining modes of $\mathcal{X}_1$ (Fig. 3.2 (f)). From this matricization, another SVD is calculated (Fig. 3.2 (g))

$$\boldsymbol{U}_2^{\langle 2 \rangle} \boldsymbol{\Sigma}_2 \boldsymbol{V}_2^{\langle d-2 \rangle^T} := \mathrm{SVD}\left(\boldsymbol{X}_1^{\langle 2 \rangle}\right) \qquad \mathcal{U}_2 \in \mathbb{R}^{r_1 \times n_2 \times r_2}, \boldsymbol{\Sigma}_2 \in \mathbb{R}^{r_2 \times r_2}, \mathcal{V}_2 \in \mathbb{R}^{n_3 \times \ldots \times n_d \times r_2} \ .$$

Again, the next step is to define

$$\boldsymbol{X}_2^{\langle 1 \rangle} := \boldsymbol{\Sigma}_2 \boldsymbol{V}_2^{\langle d-2 \rangle^T} \qquad \mathcal{X}_2 \in \mathbb{R}^{r_2 \times n_3 \times \ldots \times n_d} \ .$$

As in the first step, it holds that

$$\mathcal{U}_2 \circ \mathcal{X}_2 = \mathrm{Mat}^{-1}\left(\boldsymbol{U}_2^{\langle 2 \rangle} \boldsymbol{X}_2^{\langle 1 \rangle}\right) = \mathrm{Mat}^{-1}\left(\boldsymbol{U}_2^{\langle 2 \rangle} \boldsymbol{\Sigma}_2 \boldsymbol{V}_2^{\langle d-2 \rangle^T}\right) = \mathcal{X}_1$$

$$\Rightarrow \mathcal{U}_1 \circ \mathcal{U}_2 \circ \mathcal{X}_2 = \mathcal{X} \ .$$

The obtained rank r_2 is equal to the rank of the matricization $\boldsymbol{X}^{\langle 2\rangle}$, which can be shown as follows. Using Lemma 3.19, we know $\mathcal{W} := \mathcal{U}_1 \circ \mathcal{U}_2$, $\boldsymbol{W}^{\langle 2\rangle} \in \mathbb{R}^{n_1 n_2 \times r_2}$ is an orthogonal matrix. Now, consider that

$$\boldsymbol{X}^{\langle 2\rangle} = \mathrm{Mat}_{(1,2)}\left(\mathcal{U}_1 \circ \mathcal{U}_2 \circ \mathcal{X}_2\right) = \boldsymbol{W}^{\langle 2\rangle} \boldsymbol{\Sigma}_2 \boldsymbol{V}_2^{\langle d-2\rangle^T}$$

holds and forms a valid SVD of $\boldsymbol{X}^{\langle 2\rangle}$ with rank r_2. Since the matrix of singular values is unique, it follows that indeed $\mathrm{rank}(\boldsymbol{X}^{\langle 2\rangle}) = r_2$. This procedure is continued for a total of $d-2$ steps and in each step, the order of $\mathcal{X}_i \in \mathbb{R}^{r_i \times n_{i+1} \times \dots \times n_d}$ shrinks by one. By induction, it follows that for $1 < \mu < d-1$,

$$\mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_\mu \circ \mathcal{X}_\mu = \mathcal{X} \qquad\qquad \mathcal{U}_\mu \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}$$

and $\mathrm{rank}\left(\boldsymbol{X}^{\langle \mu\rangle}\right) = r_\mu$ hold. The $(d-1)$-st step (Fig. 3.2 (k)),

$$\boldsymbol{U}_{d-1} \boldsymbol{\Sigma}_{d-1} \boldsymbol{V}_{d-1}^T = \mathrm{SVD}\left(\boldsymbol{M}_{(1,2)}(\mathcal{X}_{d-1})\right) \quad \boldsymbol{U}_{d-1} \in \mathbb{R}^{(r_{d-2} \cdot n_{d-1}) \times r_{d-1}}, (\boldsymbol{\Sigma}_2 \boldsymbol{V}_2^T) \in \mathbb{R}^{r_{d-1} \times n_d} \ ,$$

is special, since

$$\boldsymbol{U}_d^{\langle 1\rangle} := \boldsymbol{\Sigma}_d \boldsymbol{V}_d^{\langle 1\rangle^T} \quad \mathcal{U}_d \in \mathbb{R}^{r_{d-1} \times n_d}$$

yields an order two tensor that is named $\mathcal{U}_d$ instead of $\mathcal{X}_d$ (Fig. 3.2 (l)-(m)). Finally,

$$\mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_{d-1} \circ \mathcal{U}_d = \mathcal{X}$$

is a valid TT representation of $\mathcal{X}$ with rank $\boldsymbol{r} = (r_1, \ldots, r_{d-1})$, whose entries are exactly the ranks of the matricizations $\boldsymbol{X}^{\langle \mu\rangle}$ as asserted. The orthogonality of $\boldsymbol{U}_1^{\langle 1\rangle}$ and $\boldsymbol{U}_2^{\langle 2\rangle}, \ldots, \boldsymbol{U}_{d-1}^{\langle 2\rangle}$ is evident from their definition as components of the SVDs. □

Corollary 3.21. *For every tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, the tensor train rank $\boldsymbol{r} = (r_1, \ldots, r_{d-1})$ is given by*

$$r_\mu = \mathrm{rank}\left(\boldsymbol{X}^{\langle \mu\rangle}\right) \ .$$

Proof. Theorem 3.20 shows that a TT representation with the above matricization ranks can always be obtained. It remains to show that they are also a lower bound for the corresponding entry of the TT-rank, i.e. that there cannot be a TT representation with at least one entry of the rank being smaller. Assume towards contradiction that for $\text{TT-rank}(\mathcal{X}) = (r_1, \ldots, r_{d-1})$,

$$r_\mu < \mathrm{rank}\left(\mathrm{Mat}_{(1,2,\ldots,\mu)}(\mathcal{X})\right)$$

holds for any $\mu \in \{1, \ldots, d-1\}$. Then, by definition of the TT-rank, there exist tensors $\mathcal{U}_1 \in \mathbb{R}^{n_1 \times r_1}$, $\mathcal{U}_i \in \mathbb{R}^{r_{i-1} \times n_i \times r_i}$,$\mathcal{U}_d \in \mathbb{R}^{r_{d-1} \times n_d}$, such that

$$\mathcal{X} = \mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_{d-1} \circ \mathcal{U}_d = (\mathcal{U}_1 \circ \ldots \circ \mathcal{U}_\mu) \circ (\mathcal{U}_{\mu+1} \circ \ldots \circ \mathcal{U}_d) = \mathcal{A} \circ \mathcal{B} \ ,$$

with $\mathcal{A} \in \mathbb{R}^{n_1 \times \ldots \times n_\mu \times r_\mu}$ and $\mathcal{B} \in \mathbb{R}^{r_\mu \times n_{\mu+1} \times \ldots \times n_d}$. Then,

$$\boldsymbol{X}^{\langle \mu \rangle} = \boldsymbol{A}^{\langle \mu \rangle} \boldsymbol{B}^{\langle 1 \rangle}$$

is a rank r_μ factorization of $\boldsymbol{X}^{\langle \mu \rangle}$, in contradiction to the assumption that $\operatorname{rank}(\boldsymbol{X}^{\langle \mu \rangle}) > r_\mu$. □

A modified TT-SVD algorithm can also be used to calculate a rank $\boldsymbol{r}^* = (r_1^*, \ldots, r_{d-1}^*)$ approximation of a tensor $\mathcal{X}$ with TT-rank $\boldsymbol{r} \succeq \boldsymbol{r}^*$. To this end, the normal SVDs are replaced by truncated rank r_i^* SVDs, yielding a tensor $\mathcal{X}^*$ of TT-rank $\boldsymbol{r}^*$. The following theorem shows that $\mathcal{X}^*$ is a so-called quasi best rank $\boldsymbol{r}^*$ approximation of $\mathcal{X}$.

Theorem 3.22. *Given a rank-tuple $\boldsymbol{r}^* = (r_1^*, \ldots, r_{d-1}^*)$, for every tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ the tensor $\mathcal{X}^* = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d$ obtained by the TT-SVD with all matrix SVDs replaced with rank r_μ^* truncated SVDs, fulfills*

$$\|\mathcal{X} - \mathcal{X}^*\|_F \leq \sqrt{d-1} \min_{\mathcal{Y} \in \mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}^*}\left(\mathbb{R}^{n_1 \times \ldots \times n_d}\right)} \left(\|\mathcal{X} - \mathcal{Y}\| \right) ,$$

where

$$\mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}^*}\left(\mathbb{R}^{n_1 \times \ldots \times n_d}\right) := \left\{ \mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d} \mid \text{TT-rank}\,(\mathcal{X}) \preceq \boldsymbol{r}^* \right\}$$

is the set of tensors with rank at most $\boldsymbol{r}^*$.

Proof. A proof is available in the work of Oseledets [20] and more detailed in Oseledets and Tyrtyshnikov [86] and is not repeated here. However, in Section 4.3, we present a different proof for the quasi optimality of the randomized TT-SVD introduced in that section, which can be transformed into a proof for the present theorem with only a few obvious modifications. □

As for the canonical and Tucker format, finding the true best approximation of $\mathcal{X}$ is NP-hard in general, because for rank one, i.e. $(1, \ldots, 1)$, the TT and the CP-rank coincide. Determining the best CP-rank one approximation is shown to be NP-hard by Hillar and Lim [84]. However, in contrast to the canonical format, the best approximation in the TT-format is well defined for all ranks, since the set of tensors with TT-rank at most $\boldsymbol{r}'$ is closed, as shown in the next section.

3.4.3 The Set of Tensors with Fixed Tensor Train Rank

As with matrices, the set

$$\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1 \times \ldots \times n_d}) := \left\{ \mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d} \mid \text{TT-rank}\,(\mathcal{X}) = \boldsymbol{r} \right\}$$

of tensors with fixed TT-rank is not closed. However, the set $\mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ of tensors with bounded TT-rank is closed. This can be shown by noting that matricizations are isomorphisms and therefore, the sets

$$\mathcal{M}_{\mu,r_\mu}(\mathbb{R}^{n_1\times\ldots\times n_d}) := \left\{\mathcal{X}\in\mathbb{R}^{n_1\times\ldots\times n_d} \;\middle|\; \operatorname{rank}\left(\boldsymbol{X}^{\langle\mu\rangle}\right)\le r_\mu\right\} ,$$

as pre-images of the closed matrix sets $\mathcal{M}_{\le r}(\mathbb{R}^{n_1\cdot\ldots\cdot n_{\mu-1}\times n_\mu\cdot\ldots\cdot n_d})$ from Section 3.1.4, are closed as well. Expressing

$$\mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d}) = \bigcap_{\mu=1}^{d}\mathcal{M}_{\mu,\le r_\mu}(\mathbb{R}^{n_1\times\ldots\times n_d})$$

as intersection of these sets directly shows that it is closed as well. Observing that for every rank entry, there are arbitrarily small distortions which increase that rank entry, it is clear that $\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ is dense in $\mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$, which therefore is its closure.

Analogously to the Tucker case and as a natural generalization of Section 3.1.4, it is shown by Holtz et al. [87] that the set $\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ of tensors with fixed TT rank forms a manifold. In particular, they prove the following theorem.

Theorem 3.23 (Tensor Train Manifold [87, Lemma 4.1 and Theorem 4.2])**.** *For fixed dimensions $n_1,\ldots,n_d$ and a fixed rank $\boldsymbol{r}=(r_1,\ldots,r_{d-1})$, the set $\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ is a smooth manifold of dimension*

$$\dim\left(\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})\right) = \sum_{\mu=1}^{d} r_{\mu-1}n_\mu r_\mu - \sum_{\mu=1}^{d-1} r_\mu^2 ,$$

where as a convention $r_0=r_d=1$.

Additionally, a characterization of the tangent space of this manifold is available, as given in the following.

Theorem 3.24 (Tensor Train Tangent Space (adapted from [87]))**.** *The tangent space $\mathbb{T}_{\mathcal{X}}\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ at a point $\mathcal{X}=\mathcal{U}_1\circ\ldots\circ\mathcal{U}_d\in\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ is given by*

$$\mathbb{T}_{\mathcal{X}}\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d}) = V_1\oplus\ldots\oplus V_d ,$$

where

$$V_1 = \{\mathcal{K}\circ\mathcal{U}_2\circ\ldots\circ\mathcal{U}_d \mid \mathcal{K}\in\mathbb{R}^{n_1\times r_\mu},\ \mathcal{U}_1\circ\mathcal{K}=\boldsymbol{0}\} ,$$
$$V_\mu = \left\{\mathcal{U}_1\circ\ldots\circ\mathcal{U}_{\mu-1}\circ\mathcal{K}\circ\mathcal{U}_{\mu+1}\circ\ldots\circ\mathcal{U}_d \mid \mathcal{K}\in\mathbb{R}^{r_{\mu-1}\times n_\mu\times r_\mu},\ \mathcal{U}_\mu *_{(1,2),(1,2)}\mathcal{K}=\boldsymbol{0}\right\}$$

for $\mu=2,\ldots,d-1$ and

$$V_d = \{\mathcal{U}_1\circ\ldots\circ\mathcal{U}_{d-1}\circ\mathcal{K} \mid \mathcal{K}\in\mathbb{R}^{r_{d-1}\times n_d}\} .$$

Here, it is assumed that the decomposition is normalized, such that $\mathcal{U}_1,\ldots,\mathcal{U}_{d-1}$ are left orthogonal, i.e. the core position is d.

The decomposition of the tangent space into these orthogonal subspaces has the advantage that explicit projectors onto these spaces can be given, as shown by Lubich et al. [88].

Theorem 3.25 (Tangent Subspace Projection [88]). *The projectors $\hat{P}_{V_\mu}$ onto the partition spaces V_μ of the tangent space $\mathbb{T}_{\mathcal{X}}\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}\left(\mathbb{R}^{n_1\times\ldots\times n_d}\right)$ at a point $\mathcal{X}\in\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}\left(\mathbb{R}^{n_1\times\ldots\times n_d}\right)$ from Theorem 3.24 can be given explicitly. For this, define the tensors*

$$\begin{aligned}\mathcal{U}_{<\mu} &= \mathcal{U}_1\circ\mathcal{U}_2\circ\ldots\circ\mathcal{U}_{\mu-1},\\ \mathcal{U}_{>\mu} &= \mathcal{U}_{\mu+1}\circ\mathcal{U}_{\mu+2}\circ\ldots\circ\mathcal{U}_d.\end{aligned}$$

Here, the $\mathcal{U}_k$ are the left and right orthogonal components of a TT representation of $\mathcal{X}$, respectively, such that the core is at position μ. For $\mu = 1,\ldots,d-1$, the projector is composed of

$$\hat{P}_{V_\mu} = \hat{P}^+_\mu - \hat{P}^-_\mu\ ,$$

where the $\hat{P}^+_\mu$ project onto the spaces

$$\begin{aligned}\tilde{V}_1 &= \left\{\mathcal{K}\circ\mathcal{U}_2\circ\ldots\circ\mathcal{U}_d \mid \mathcal{K}\in\mathbb{R}^{n_\mu\times r_\mu}\right\}\ ,\\ \tilde{V}_\mu &= \left\{\mathcal{U}_1\circ\ldots\circ\mathcal{U}_{\mu-1}\circ\mathcal{K}\circ\mathcal{U}_{\mu+1}\circ\ldots\circ\mathcal{U}_d \mid \mathcal{K}\in\mathbb{R}^{r_{\mu-1}\times n_\mu\times r_\mu}\right\}\ ,\end{aligned}$$

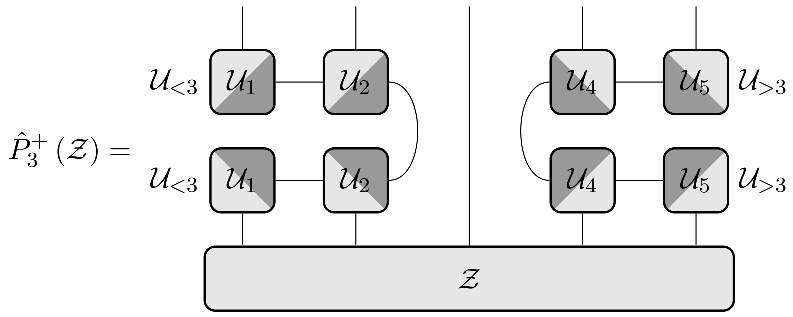
and $\hat{P}^-_\mu$ enforces the orthogonality constraint. For $\mu = d$, there is no such constraint and $\hat{P}_{V_\mu} = \hat{P}^+_\mu$ holds. The partial projectors are defined as

$$\hat{P}^+_\mu(\mathcal{Z}) = \Big(\mathcal{U}_{<\mu} *_{(\mu),(\mu)} \mathcal{U}_{<\mu}\Big) *_{(1,\ldots,\mu-1),(1,\ldots,\mu-1)} \mathcal{Z} *_{(\mu+1,\ldots,d),(1,\ldots,d-\mu)} \Big(\mathcal{U}_{>\mu} *_{(1),(1)} \mathcal{U}_{>\mu}\Big).$$

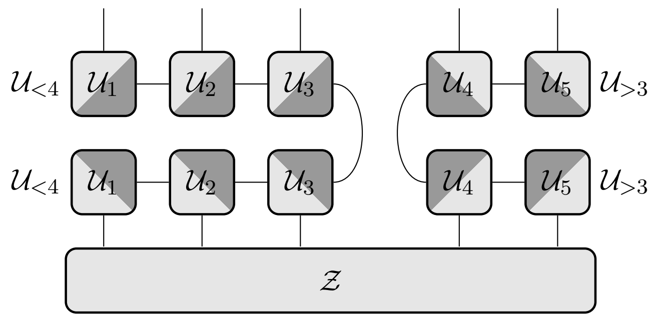
and

$$\hat{P}^-_\mu(\mathcal{Z}) = \Big\{\mathcal{U}_{<\mu+1} *_{(\mu+1),(\mu+1)} \mathcal{U}_{<\mu+1}\Big) *_{(1,\ldots,\mu),(1,\ldots,\mu)} \mathcal{Z} *_{(\mu+1,\ldots,d),(1,\ldots,d-\mu)} \Big(\mathcal{U}_{>\mu} *_{(1),(1)} \mathcal{U}_{>\mu}\Big\}\ .$$

The composition of the projector can be expressed intuitively in the graphical notation, shown for an example of an order five tensor in the following.



$\hat{P}_3^-(\mathcal{Z}) =$



These subspace projectors will play an important role in the formulation of optimization algorithms in the later chapters of this thesis.

3.4.4 Computational Aspects

To store a tensor $\mathcal{X} \in \mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ in the tensor train decomposition, it is sufficient to store the d component tensors $\mathcal{U}_\mu$. Setting $n = \max(n_1, \dots, n_d)$ and $r = \max(r_1, \dots, r_d)$, this requires order $\Theta(dnr^2)$ memory. For $r \ll n$, this offers a tremendous reduction compared to the normal $\Theta(n^d)$. In particular, the TT format realizes a linear scaling with respect to the order d. A similar scaling can also be achieved for common operations, if all operands are given in a low rank tensor train representation. This makes the tensor train representation the most versatile format introduced so far.

Analogously to the previous discussions of computational aspects for matrices, and the CP- and Tucker formats, in Sections 3.1.5, 3.2.4 and 3.3.4, respectively, we will now discuss the complexity of various operations in the TT format. In the following, assume that $\mathcal{X}, \bar{\mathcal{X}} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ are two tensors of TT-ranks $r = (r_1, \dots, r_{d-1})$ and $\bar{r} = (\bar{r}_1, \dots, \bar{r}_{d-1})$, respectively. Apart from the first paragraph, we assume that both tensors are given as TT decompositions

$$\begin{aligned} \mathcal{X} &= \mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_d \\ \bar{\mathcal{X}} &= \bar{\mathcal{U}}_1 \circ \bar{\mathcal{U}}_2 \circ \ldots \circ \bar{\mathcal{U}}_d \ . \end{aligned}$$

For all statements on the computational complexity, define $n = \max(n_1, \dots, n_d)$, $r = \max(r_1, \dots, r_{d-1})$ and $\bar{r} = \max(\bar{r}_1, \dots, \bar{r}_{d-1})$.

Normalization, (Re-)Orthogonalization Assume that $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ is given in a TT representation, but the component tensors are not necessarily orthogonalized. The reorthogonalization can be computed using QR-decompositions. Starting at $\mathcal{U}_1$, one calculates

$$\begin{aligned}
\tilde{\boldsymbol{U}}_1^{<1>}\boldsymbol{R} &:= \mathrm{QR}\Big(\boldsymbol{U}_1^{<1>}\Big) \\
\mathcal{W}_2 &:= \boldsymbol{R} \circ \mathcal{U}_2 \\
\tilde{\boldsymbol{U}}_2^{<2>}\boldsymbol{R} &:= \mathrm{QR}\Big(\boldsymbol{W}_2^{<2>}\Big) \\
\mathcal{W}_3 &:= \boldsymbol{R} \circ \mathcal{U}_3 \\
&\vdots \\
\tilde{\boldsymbol{U}}_{d-1}^{<2>}\boldsymbol{R} &:= \mathrm{QR}\Big(\boldsymbol{W}_{d-1}^{<2>}\Big) \\
\tilde{\mathcal{U}}_d &:= \boldsymbol{R} \circ \mathcal{U}_d
\end{aligned}$$

At the end, the core is at the d-th node and all $\tilde{\mathcal{U}}_1, \ldots, \tilde{\mathcal{U}}_{d-1}$ are left-orthogonal, as required. Note that because $\boldsymbol{R}$ is always a $r_\mu \times r_\mu$ matrix, this procedure cannot increase the rank entries. At the same time, the representation obtained may possibly have a non-minimal rank. The computational cost scales as $\mathcal{O}(dnr^3)$.

Move core The core position can be changed by calculating the QR or RQ decomposition of the current core component. In particular, to increase the core position from μ to $\mu+1$, calculate

$$\boldsymbol{Q}^{<2>}\boldsymbol{R} := \mathrm{QR}\Big(\boldsymbol{U}_\mu^{<2>}\Big)$$

and set $\boldsymbol{U}_\mu^{<2>} := \boldsymbol{Q}^{<2>}$ and $\boldsymbol{U}_{\mu+1}^{<1>} := \boldsymbol{R}\boldsymbol{U}_{\mu+1}^{<1>}$. Now, $\mathcal{U}_\mu$ is left-orthogonal and $\mathcal{U}_{\mu+1}$ carries the core, as required. All other components remain unchanged. Analogously, to decrease the core position from μ to $\mu - 1$, calculate

$$\boldsymbol{R}\boldsymbol{Q}^{<1>} := \mathrm{RQ}\Big(\boldsymbol{U}_\mu^{<1>}\Big)$$

and set $\boldsymbol{U}_\mu^{<1>} := \boldsymbol{Q}^{<1>}$ and $\boldsymbol{U}_{\mu-1}^{<2>} := \boldsymbol{U}_{\mu-1}^{<2>}\boldsymbol{R}$. The computational cost for one step scales as $\mathcal{O}(nr^3)$. By repeating these steps, any core position can be reached.

Rank Truncation To reduce the ranks to the optimal TT-rank, or to obtain a quasi-best rank $\boldsymbol{r}' \prec \boldsymbol{r}$ approximation of $\mathcal{X}$, a sweep of matrix SVDs is required. This assumes that the representation is already orthogonalized with the core position one, otherwise a prior orthogonalization as described above is required. The algorithm works as follows:

$$\begin{aligned}
\tilde{\boldsymbol{U}}_1^{<1>}\boldsymbol{\Sigma}_{r_1'}\boldsymbol{V}_1^T &:= \mathrm{SVD}_{r_1'}\Big(\boldsymbol{U}_1^{<1>}\Big)\\
\mathcal{W}_2 &:= \Big(\boldsymbol{\Sigma}_{r_1'}\boldsymbol{V}_1^T\Big)\circ\mathcal{U}_2\\
\tilde{\boldsymbol{U}}_2^{<2>}\boldsymbol{\Sigma}_{r_2'}\boldsymbol{V}_2^T &:= \mathrm{SVD}_{r_2'}\Big(\boldsymbol{W}_2^{<2>}\Big)\\
\mathcal{W}_3 &:= \Big(\boldsymbol{\Sigma}_{r_2'}\boldsymbol{V}_2^T\Big)\circ\mathcal{U}_3\\
&\vdots\\
\tilde{\boldsymbol{U}}_{d-1}^{<2>}\boldsymbol{\Sigma}_{r_{d-1}'}\boldsymbol{V}_{d-1}^T &:= \mathrm{SVD}_{r_{d-1}'}\Big(\boldsymbol{W}_{d-1}^{<2>}\Big)\\
\tilde{\mathcal{U}}_d &:= \Big(\boldsymbol{\Sigma}_{r_{d-1}'}\boldsymbol{V}_{d-1}^T\Big)\circ\mathcal{U}_d\ ,
\end{aligned}$$

where $\boldsymbol{\Sigma}_{r_\mu'}$ denotes the rank r_μ' truncated matrix of singular values. At the end, $\tilde{\mathcal{U}}_1\circ\ldots\circ\tilde{\mathcal{U}}_d$ is a $\boldsymbol{r}'$ approximation of $\mathcal{X}$ with core position d. Due to the inital orthogonality of $\mathcal{U}_2,\ldots,\mathcal{U}_d$, at each step $\mathcal{W}_\mu$ carries the core and due to the uniqueness of the singular values, it holds that

$$\begin{aligned}
&\tilde{\boldsymbol{U}}_\mu^{<2>}\boldsymbol{\Sigma}_\mu\boldsymbol{V}^T := \mathrm{SVD}\Big(\boldsymbol{W}_\mu^{<2>}\Big)\Rightarrow\\
&\underbrace{\mathrm{Mat}_{1,\ldots,\mu}\Big(\tilde{\mathcal{U}}_1\circ\ldots\tilde{\mathcal{U}}_\mu\Big)}_{\boldsymbol{Q}_L}\boldsymbol{\Sigma}_\mu\underbrace{\boldsymbol{V}^T\,\mathrm{Mat}_1(\mathcal{U}_{\mu+1}\circ\ldots\mathcal{U}_d)}_{\boldsymbol{Q}_R^T}\\
&=\mathrm{SVD}\Big(\mathrm{Mat}_{1,\ldots,\mu}\Big(\tilde{\mathcal{U}}_1\circ\ldots\circ\tilde{\mathcal{U}}_{\mu-1}\circ\mathcal{W}_\mu\circ\mathcal{U}_{\mu+1}\circ\ldots\circ\mathcal{U}_d\Big)\Big)\ .
\end{aligned}$$

It follows that the approximations done by this procedure are exactly the ones of the TT-SVD and therefore, the quasi best approximation properties of the TT-SVD apply. However, the computational complexity scales only as $\mathcal{O}(dnr^3)$, instead of $\mathcal{O}(n^{d+1}+dn^dr^2)$ for the TT-SVD.

Access Entries The entries $\mathcal{X}[i_1,\ldots,i_d]$ cannot be accessed directly in the Tensor Train representation. Using the matrix notation (3.13) for the TT-format, the entries are obtained from the matrix products

$$\mathcal{X}[i_1,\ldots,i_d]=\boldsymbol{U}_{1,i_1}\boldsymbol{U}_{2,i_2}\ldots\boldsymbol{U}_{d-1,i_{d-1}}\boldsymbol{U}_{d,i_d}\ .$$

The computational cost to calculate this for one entry scales as $\mathcal{O}(dr^2)$, compared to the constant access cost for directly stored tensors.

Addition Similar to the methods for matrices, the sum $\mathcal{X}+\bar{\mathcal{X}}$ can be computed while remaining in a low rank representation at all times. Using the matrix notation (3.13) for the TT-format, one can express $\mathcal{X}+\bar{\mathcal{X}}$ as

$$\begin{aligned}
&\left(\mathcal{X}+\bar{\mathcal{X}}\right)[i_1,\dots,i_d]\\
&=\boldsymbol{U}_{1,i_1}\,\boldsymbol{U}_{2,i_2}\dots\boldsymbol{U}_{d-1,i_{d-1}}\boldsymbol{U}_{d,i_d}+\bar{\boldsymbol{U}}_{1,i_1}\,\bar{\boldsymbol{U}}_{2,i_2}\dots\bar{\boldsymbol{U}}_{d-1,i_{d-1}}\,\bar{\boldsymbol{U}}_{d,i_d}\\
&=\begin{pmatrix}\boldsymbol{U}_{1,i_1} & \bar{\boldsymbol{U}}_{1,i_1}\end{pmatrix}\begin{pmatrix}\boldsymbol{U}_{2,i_2} & 0\\ 0 & \bar{\boldsymbol{U}}_{2,i_2}\end{pmatrix}\dots\begin{pmatrix}\boldsymbol{U}_{d-1,i_{d-1}} & 0\\ 0 & \bar{\boldsymbol{U}}_{d-1,i_{d-1}}\end{pmatrix}\begin{pmatrix}\boldsymbol{U}_{d,i_d}\\ \bar{\boldsymbol{U}}_{d,i_d}\end{pmatrix}\\
&=\boldsymbol{W}_{1,i_1}\,\boldsymbol{W}_{2,i_2}\dots\boldsymbol{W}_{d-1,i_{d-1}}\,\boldsymbol{W}_{d,i_d}\ .
\end{aligned}\tag{3.15}$$

The $\boldsymbol{W}_{k,i}$ are $(r_{k-1}+\bar{r}_{k-1})\times(r_k+\bar{r}_k)$ matrices, thus (3.15) is a valid TT representation of $\left(\mathcal{X}+\bar{\mathcal{X}}\right)$ of rank $\boldsymbol{r}+\bar{\boldsymbol{r}}$. The computational costs for the addition itself scale as $\mathcal{O}(dn(r+\bar{r})^2)$, compared to $\mathcal{O}(n^d)$ for directly stored tensors. Note that in general, the representation obtained is not of minimal rank, and requires a subsequent reorthogonalization.

Hadamard Product The entrywise product of $\mathcal{X}$ and $\bar{\mathcal{X}}$ can be computed within a low rank representation. To this end, define the tensors $\mathcal{W}_1\in\mathbb{R}^{n_1\times r_1\bar{r}_1}$, $\mathcal{W}_\mu\in\mathbb{R}^{r_{\mu-1}\bar{r}_{\mu-1}\times n_\mu\times r_\mu\bar{r}_\mu}, 1<\mu<d$ and $\mathcal{W}_d\in\mathbb{R}^{r_{d-1}\bar{r}_{d-1}\times n_d}$ as

$$\begin{aligned}
\mathcal{W}_1[i_1,k_1] &:= \mathcal{U}_1[i_1,k_1\ \mathbf{div}\ \bar{r}_1]\ \bar{\mathcal{U}}_1[i_1,k_1\ \mathbf{mod}\ \bar{r}_1]\\
\mathcal{W}_\mu[k_{\mu-1},i_\mu,k_\mu] &:= \mathcal{U}_\mu[k_{\mu-1}\ \mathbf{div}\ \bar{r}_{\mu-1},i_\mu,k_\mu\ \mathbf{div}\ \bar{r}_\mu]\ \bar{\mathcal{U}}_\mu[k_{\mu-1}\ \mathbf{mod}\ \bar{r}_{\mu-1},i_\mu,k_\mu\ \mathbf{mod}\ \bar{r}_\mu]\\
\mathcal{W}_d[k_{d-1},i_d] &:= \mathcal{U}_d[k_{d-1}\ \mathbf{div}\ \bar{r}_{d-1},i_d]\ \bar{\mathcal{U}}_d[k_{d-1}\ \mathbf{mod}\ \bar{r}_{d-1},i_d]\ ,
\end{aligned}$$

where **div** and **mod** denote the integer division and modulus, respectively. The Hadamard product can then be given as

$$\begin{aligned}
&\left(\mathcal{X}\odot\bar{\mathcal{X}}\right)[i_1,\dots,i_d]\\
&=(\mathcal{U}_1\circ\mathcal{U}_2\circ\dots\circ\mathcal{U}_d)[i_1,\dots,i_d]\ \left(\bar{\mathcal{U}}_1\circ\bar{\mathcal{U}}_2\circ\dots\circ\bar{\mathcal{U}}_d\right)[i_1,\dots,i_d]\\
&=\sum_{p_1=1}^{r_1}\cdots\sum_{p_{d-1}=1}^{r_{d-1}}\sum_{q_1=1}^{\bar{r}_1}\cdots\sum_{q_{d-1}=1}^{\bar{r}_{d-1}}\\
&\quad\mathcal{U}_1[i_1,p_1]\ \mathcal{U}_2[p_1,i_2,p_2]\dots\mathcal{U}_d[p_{d-1},i_d]\ \bar{\mathcal{U}}_1[i_1,q_1]\ \bar{\mathcal{U}}_2[q_1,i_2,q_2]\dots\bar{\mathcal{U}}_d[q_{d-1},i_d]\\
&=\sum_{k_1=1}^{r_1\bar{r}_1}\cdots\sum_{k_{d-1}=1}^{r_{d-1}\bar{r}_{d-1}}\mathcal{W}_1[i_1,k_1]\ \mathcal{W}_2[k_1,i_2,k_2]\dots\mathcal{W}_d[k_{d-1},i_d]\ ,
\end{aligned}$$

which is a rank $\boldsymbol{r}+\bar{\boldsymbol{r}}$ representation. Note that the representation is in general neither orthogonalized, nor of minimal rank. The computational cost scales as $\mathcal{O}(dnr^2\bar{r}^2)$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

Frobenius Inner Product The Frobenius scalar product of $\mathcal{X}$ and $\bar{\mathcal{X}}$ can be calculated via

$$\left\langle\mathcal{X},\bar{\mathcal{X}}\right\rangle_F=\left(\left(\cdots\left(\left(\left(\left(\mathcal{U}_1*_{1,1}\bar{\mathcal{U}}_1\right)*_{1,1}\mathcal{U}_2\right)\circ_2\bar{\mathcal{U}}_2\right)*_{1,1}\mathcal{U}_3\right)\circ_2\bar{\mathcal{U}}_3\dots\right)*_{1,1}\mathcal{U}_d\right)\circ_2\bar{\mathcal{U}}_d\ .$$

Performing the contractions in this order incurs a computational cost of $\mathcal{O}(dnr^3)$, compared to $\mathcal{O}(n^d)$ for directly stored tensors.

Frobenius Norm Assuming that the representation is orthogonalized and that the core is at the μ-th node, the Frobenius norm can be calculated as

$$\|\mathcal{X}\|_F = \|\mathcal{U}_\mu\|_F \ .$$

The computational complexity scales as $\mathcal{O}(rn)$ (if μ equals 1 or d) respectively $\mathcal{O}(r^2 n)$ (otherwise), compared to $\mathcal{O}(n^d)$ for directly stored tensors.

ν-th Mode Product Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n_\nu \times m}$, the ν-th mode product $\mathcal{X} \times_\nu \boldsymbol{A}$ can be computed as

$$\begin{aligned}\mathcal{X} \times_\nu \boldsymbol{A} &= (\mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_d) \times_\nu \boldsymbol{A} \\ &= \mathcal{U}_1 \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_{\nu-1} \circ (\mathcal{U}_\nu \times_2 \boldsymbol{A}) \circ \mathcal{U}_{\nu+1} \circ \ldots \circ \mathcal{U}_d \ .\end{aligned}$$

If ν was the core position, the representation remains orthogonalized, but in general not of minimal rank. The computational complexity scales as $\mathcal{O}(mnr^2)$, compared to $\mathcal{O}(mn^d)$ for directly stored tensors.

Other Contractions Several other contractions can be computed staying in a low rank representation at all times, if both operands are given in a TT-representation or other low rank formats. Some examples will occur later within this thesis.

3.5 Hierarchical Tucker and Other Tensor Network Decompositions

Apart from the tensor decompositions presented in the previous sections, there is quite a number of further decompositions, all based on the idea of decomposing a tensor into a specific class of tensor networks. From a mathematical point of view, the most important one is the Hierarchical Tucker (HT) decomposition, introduced by Hackbusch and Kühn [16]. The following gives a brief summary of the construction, the reader is referred to the introduction by Hackbusch and Schneider [89] for a formal definition. The fundamental idea is the same as for the Tensor Train format, but instead of using the nested subspace sequence (3.8), the final subspace is constructed from a hierarchy of subspaces. To this end, a partition tree for the set of mode-indices is defined, such that the root of the tree contains the complete set, each leaf contains a single mode-index and each inner node of the tree contains the union of its children. Two examples of partition trees for four modes (i.e. order four tensors) are given in Figure 3.3. Starting from the lowest level (furthest from the root), the HT-decomposition constructs subspaces for each node. For leaf nodes, these are the subspaces $U_\mu \subseteq \mathbb{R}^{n_\mu}$, while for the inner nodes, the subspaces are subspaces of the tensor product of the subspaces associated with the child nodes. The dimensions of these subspaces define the HT-rank tuple. To represent a tensor, which is an element of the

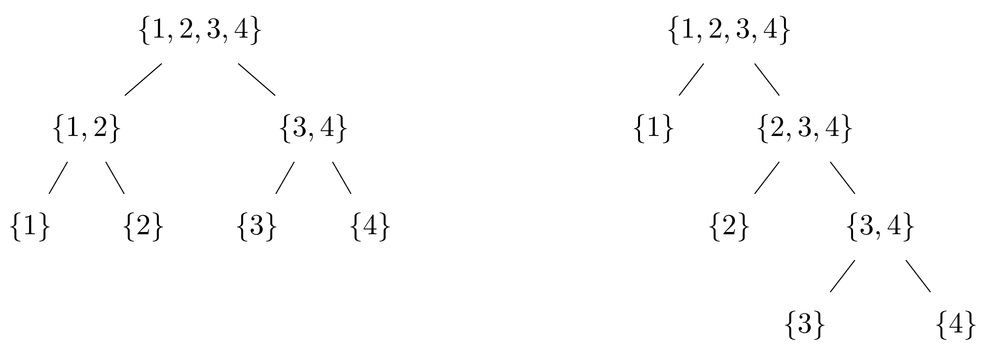


Figure 3.3: Exemplary partition trees for the Hierarchical Tucker format for order four tensors from $\mathbb{R}^{n_1\times\ldots\times n_d}$.

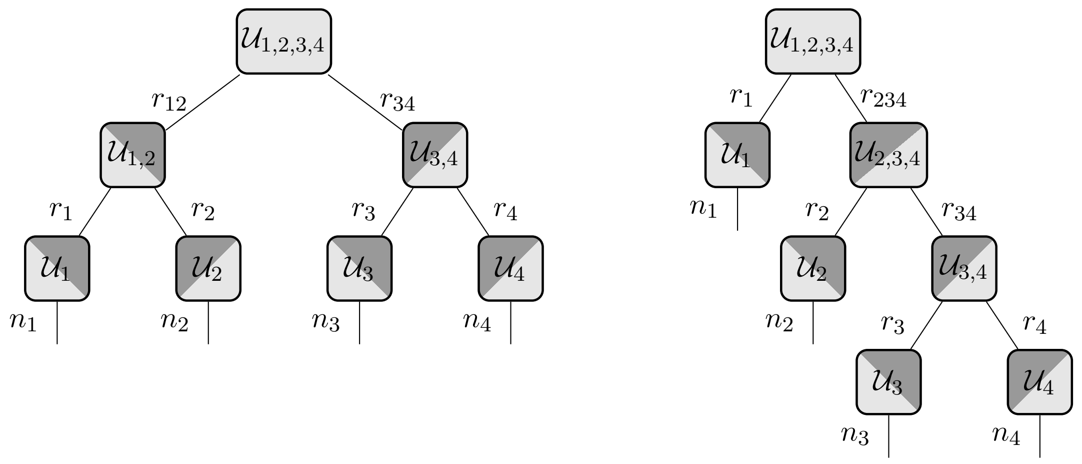


Figure 3.4: Diagrammatic notation of a balanced hierarchical Tucker decomposition (left) and the realization of a tensor train in the HT format (right), for an example of an order four Tensor. The formats correspond to the partition trees from Figure 3.3.

root subspace, it is sufficient to store the basis matrices for the leaves, the transfer-tensors, which transform the bases of the child nodes to a basis of each inner node and lastly, the coefficient root tensor. The resulting tensor networks for the two partition trees of Figure 3.3 are depicted in Figure 3.4. For a fixed hierarchical rank, the storage requirement to store these scales as $\mathcal{O}(dnr + dr^3)$, i.e. in particular only linearly in the order. Further properties of the Tucker and TT-format, including the generalized SVD and the low rank manifold, also extend to the HT-format. In the literature, it is sometimes claimed that the TT format is a special case of the HT format. This is not entirely correct, at least not for the TT format as introduced in the previous section, because the HT format requires basis matrices for every mode to be separated, as in the Tucker format (see Fig. 3.4). A detailed discussion of the differences between the TT and HT format is given by Grasedyck and Hackbusch [90], who also give an extensive introduction to the HT format in general.

Especially in quantum physics, even more general tensor networks are used. For example, there are the projected entangled pair states (PEPS), which can be seen as a generalization of MPS to higher dimension [91], the multi-scale entanglement renormalization ansatz

Action	CP	Tucker	TT	HT
SVD	-	dn^{d+1}	$n^{d+1} + dn^d r^2$	$dn^{d+1} + dn^d r^2$
Orthogonalization	-	$dr^{d+1} + dnr^2$	dnr^3	$dnr^2 + dr^4$
Rank Reduction	-	$dr^{d+1} + dnr^2$	dnr^3	$dnr^2 + dr^4$
Access Entry	dr	r^d	dr^2	dr^3
Addition*	$dn(r + \bar{r})$	$dn(r + \bar{r}) + (r + \bar{r})^d$	$dn(r + \bar{r})^2$	$dn(r + \bar{r}) + d(r + \bar{r})^3$
Hadamard Product*	$dnr\bar{r}$	$dnr\bar{r} + r^d\bar{r}^d$	$dnr^2\bar{r}^2$	$dnr\bar{r} + dr^3\bar{r}^3$
Inner Product	$dnr\bar{r}$	$dnr\bar{r} + dr\bar{r}^d + r^d$	dnr^3	$dnr^2 + dr^4$
Frobenius Norm	dnr^2	r^d	rn	r^2
μ-th Mode Product*	$(d+m)nr^2$	mnr	mnr^2	mnr
Storage	dnr	$r^d + dnr$	dnr^2	$dnr + dr^3$

Table 3.1: Comparison of the asymmtotic computational complexity of several operations for the different decomposition formats. By convention, $n = \max(n_\mu)$, $r = \max(r_\mu)$ and $\bar{r} = \max(\bar{r}_\mu)$. The values for the HT-format assume a binary partition tree. Operations marked with * may yield an unorthogonalized representation with sub-optimal rank.

(MERA) [92] and the tree tensor networks (TTN) [93]. While there are individual advantages and disadvantages for each format, there is a qualitative difference between those formats for which the tensor network is described by a tree (i.e. which is cycle free) and those for which it is not (e.g. PEPS and MERA). For example, cyclic networks can not be constructed by a generalized SVD and there is no well defined minimal rank, because any rank in a loop can be decreased by increasing the other ranks within the loop. An interesting discussion on the closedness of tensor networks containing cycles can be found in the work of Landsberg et al. [94].

3.6 Comparison

All low rank formats presented in the previous sections have unique advantages and disadvantages. Especially the scaling of different operations differs significantly, as summarized in Table 3.1. For a fixed rank, the canonical format has a superior scaling for most operations and in particular breaks the curse of dimensionality by allowing a linear scaling with respect to the order for all mentioned operations. However, the CP-format lacks a generalized SVD method to obtain a low rank approximation in the first place, as well as further obstacles described in Section 3.2. These obstacles are overcome by all three subspace decompositions, which offer a generalized higher order SVD and also allow a manifold structure on the set of low rank tensors. The Tucker format as the simplest subspace decomposition still suffers from the curse of dimensionality, exhibiting an exponential scaling with respect to the order for most mentioned operations. The Tensor Train and the Hierarchical Tucker format using an incremental hierarchy of subspaces are able to circumvent this problem, indeed allowing a linear scaling with respect to the order while inheriting the advantages of a generalized SVD and manifold structure from the Tucker format. Formally, the Tensor Train format offers a better scaling for most operations, but this neglects the fact that the meaning of rank differs for all formats. A noteworthy exception is the fact that for all formats the set of rank one tensors coincides and is equal to the set of elementary tensors. For all other tensors, however, the CP-, Tucker-, TT- and HT-rank may differ significantly. See for example the work of Grasedyck and Hackbusch [90] for some bounds between the TT- and HT-ranks. In practical applications, it is therefore important to choose the format according to the underlying structure of the problem.

4 Randomized Tensor Decomposition Methods

This chapter presents a set of randomized methods for the calculation of low rank tensor decompositions. The main results are randomized HOSVD and TT-SVD algorithms, for which stochastic error bounds are proven. It is shown that the randomized TT-SVD has a computational complexity scaling only linearly in the order if applied to sparse tensors, making it a powerful tool for the decomposition of such tensors. Parts of this work were previously published in [95] together with R. Schneider and B. Huber.

As shown in the previous chapter, calculating a low rank representation or approximation of a given higher order tensor is a challenging task. For the canonical format, the situation is worst, as there is no straightforward way to calculate a decomposition or approximation at all. Furthermore, for the Tucker and Tensor Train formats, the costs for calculating the HOSVD or TT-SVD scale exponentially in the order. For dense tensors, this is of course somewhat expected, as there is an exponential number of entries that have to be incorporated. Nevertheless, also for sparse and structured tensors, the algorithms exhibit an exponential scaling. This chapter shows that randomized methods can severely reduce the computational complexity, especially for such sparse or structured tensors. Let us highlight that the calculation of low rank decompositions and approximations is at the heart of several optimization algorithms on the low rank Tucker or Tensor Train manifolds, see for example the survey of Grasedyck et al. [46, Section 3.1] and references therein. Many of these algorithms can directly profit from the randomized approach proposed in this chapter. An example of special interest for this thesis is the iterative hard thresholding (IHT) algorithm for tensor completion, originally proposed by Rauhut et al. [61] and refined in [65, 96].

The algorithms proposed in this chapter build upon the work of Halko et al. [47], who developed randomized methods for the calculation of approximate matrix factorizations. Especially for sparse or structured matrices, these techniques allow the very efficient calculation of common matrix factorizations, such as the SVD or QR decomposition, while offering rigorous stochastic error bounds. This chapter starts with a brief summary of these results in Section 4.1. In the following sections, randomized variants of the HOSVD (Section 4.2) and TT-SVD (Section 4.3) are presented and stochastic error bounds are derived. Section 4.4 discusses the use of structured randomness and shows an interesting relation between the alternating least squares (ALS) algorithm and the randomized TT-

SVD. In Section 4.5, we provide numerical evidence for the practical applicability of the presented methods. Finally, Section 4.6 summarizes the results and gives an outlook on future work.

4.1 Randomized Singular Value Decomposition for Matrices

Randomized techniques for the calculation of singular value decompositions and QR factorizations of matrices have been proposed many times in the literature. However, it was only rather recently that these procedures could be analyzed rigorously, thanks to the application of new insights from *random matrix theory*. The stochastic error bounds presented here were originally published by Halko et al. [47]. This section focuses on the usage of standard Gaussian random matrices, i.e. matrices whose entries are i.i.d. standard Gaussian random variables. The usage of structured random matrices is briefly discussed in Section 4.4.

The fundamental method proposed by Halko et al. [47] is a randomized procedure, which calculates an approximate low rank subspace projection

$$\boldsymbol{A} \approx \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{A} \ , \tag{4.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ is a given matrix and $\boldsymbol{Q} \in \mathbb{R}^{m\times s}$ is an orthogonal matrix approximately spanning the range of $\boldsymbol{A}$. Here, $s = r + p$ is composed of the desired rank r and an oversampling parameter p. With this projection at hand, numerous different low rank decompositions can be calculated deterministically at low costs. For example, the singular value decomposition of $\boldsymbol{A}$ can be calculated by forming $\boldsymbol{B} := \boldsymbol{Q}^T\boldsymbol{A}$ and calculating the deterministic SVD $\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T = \boldsymbol{B}$. Using $\tilde{\boldsymbol{U}} = \boldsymbol{Q}\boldsymbol{U}$,

$$\tilde{\boldsymbol{U}}\boldsymbol{S}\boldsymbol{V}^T = \boldsymbol{Q}\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T = \boldsymbol{Q}\boldsymbol{B} = \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{A} \approx \boldsymbol{A}$$

is an approximate SVD of $\boldsymbol{A}$ containing only the approximation error incurred by the subspace-projection. The computational costs scale as $\mathcal{O}\left(sT_{\text{mult}} + s^2(m+n)\right)$, where T_{mult} is the cost to calculate the matrix-vector product with $\boldsymbol{A}$, which is $\mathcal{O}(mn)$ for a general matrix but can be much lower for sparse or structured matrices. In a very similar way, other matrix factorizations can be computed with low costs if the projection (4.1) is given, see [47].

The main challenge is the calculation of the approximate range $\boldsymbol{Q}$ through randomized techniques. As a motivation, consider a set of s random vectors $\boldsymbol{g}_1, \dots, \boldsymbol{g}_s \in \mathbb{R}^n$ and set $\boldsymbol{x}_i = \boldsymbol{A}\boldsymbol{g}_i$. If the matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ has rank $r \le s$, then with high probability, the set $\{\boldsymbol{x}_i \mid i = 1, \dots, s\}$ spans the range of $\boldsymbol{A}$. While the details depend on the choice of randomness for the vectors $\boldsymbol{g}_i$, the argument of the proof can be summarized as follows. First, let

$$\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \boldsymbol{A} \qquad\qquad \boldsymbol{U} \in \mathbb{R}^{m\times m}, \boldsymbol{\Sigma} \in \mathbb{R}^{m\times n}, \boldsymbol{V} \in \mathbb{R}^{n\times n}$$

be a *complete* SVD of $\boldsymbol{A}$. It is clear that with high probability, the vectors $\boldsymbol{g}_i$ are in general linear position, i.e. span a s-dimensional subspace of $\mathbb{R}^n$ (unless $s > n$, in which case they span the entire space). Applying the orthogonal matrix $\boldsymbol{V}^T$ does not change this and so the set $\{\boldsymbol{y}_i = \boldsymbol{V}^T\boldsymbol{g}_i \mid i = 1, \ldots, s\}$ is in general linear position as well. Applying $\boldsymbol{\Sigma}$ sets the last $n - r$ entries of each vector to zero. As the entries are random, the truncated vectors still span a $\min(r, s)$-dimensional subspace with high probability. The scaling of the first r values through the application of $\boldsymbol{\Sigma}$ does not break the linear independence. Consequently, with high probability, $\{\boldsymbol{z}_i = \boldsymbol{\Sigma}\boldsymbol{x}_i \mid i = 1, \ldots, s\}$ spans a $\min(r, s)$-dimensional subspace of $\mathbb{R}^m$. The application of the orthogonal matrix $\boldsymbol{U}$ again preserves linear independence and with high probability, $\{\boldsymbol{x}_i = \boldsymbol{U}\boldsymbol{z}_i = \boldsymbol{A}\boldsymbol{g}_i \mid i = 1, \ldots, s\}$ spans a $\min(r, s)$-dimensional subspace of $\mathbb{R}^m$. In this case, $\{\boldsymbol{x}_i \mid i = 1, \ldots, s\}$ spans the complete range of $\boldsymbol{A}$ if $s \geq r$, because by construction $\boldsymbol{x}_i \in \operatorname{range}(\boldsymbol{A})$, and the dimension of the range is equal to the rank. To obtain the matrix $\boldsymbol{Q}$ as above, one may construct an orthogonal basis of this span, e.g. by calculating the QR decomposition $\boldsymbol{QR} = \boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s)$, which is comparatively inexpensive if $s \ll n$. Algorithm 4 formalizes this prototype algorithm using an equivalent random matrix instead of the random vectors $\boldsymbol{g}_i$. The remarkable result of Halko et al. [47], given in the following theorem, is that this approach also provides a good approximation of the range for $r > s$, in the sense of (4.1).

Algorithm 4: Randomized Matrix Range Approximation

```
Input  : Target $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, sampling parameter $s \in \mathbb{N}_+$
Output : Approximated range $\boldsymbol{Q} \in \mathbb{R}^{m\times s}$
1 Create a standard Gaussian random matrix $\mathbf{G} \in \mathbb{R}^{n\times s}$
2 Calculate the intermediate matrix $\boldsymbol{B} := \mathbf{AG} \in \mathbb{R}^{m\times s}$.
3 Compute the QR-factorization $\boldsymbol{QR} = \boldsymbol{B}$.
```

Theorem 4.1 (Halko et al. [47, Theorem 10.7 and Theorem 10.8])**.** *Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ be a matrix, let r be a target rank and let $p \geq 4$ an oversampling parameter with $s := r + p$. Then the following error bounds hold for the orthogonal projection $\boldsymbol{QQ}^T$ obtained by Algorithm 4. For all $u, t \geq 1$,*

$$\|\mathbf{A} - \mathbf{QQ}^T\mathbf{A}\|_F \leq \left(1 + t\sqrt{\frac{12r}{p}}\right)\left(\sum_{k>r}\sigma_k(\boldsymbol{A})^2\right)^{1/2} + ut\frac{e\sqrt{r+p}}{p+1}\sigma_{r+1}(\boldsymbol{A})$$

and

$$\|\mathbf{A} - \mathbf{QQ}^T\mathbf{A}\|_2 \leq \left(1 + t\sqrt{\frac{12r}{p}}\right)\sigma_{r+1}(\boldsymbol{A}) + t\frac{e\sqrt{r+p}}{p+1}\left(\sum_{k>r}\sigma_k(\boldsymbol{A})^2\right)^{1/2} + ut\frac{e\sqrt{r+p}}{p+1}\sigma_{r+1}(\boldsymbol{A})$$

each hold with probability at least $1 - 5t^{-p} - 2e^{-u^2/2}$. Here, $\sigma_i(\boldsymbol{A})$ denotes the i-th singular value of $\boldsymbol{A}$.

Note that if the matrix $\boldsymbol{A}$ has rank r or less, that is, if $\sigma_{r+1} = \sigma_{r+2} = \ldots = 0$, then the subspace projection is exact with probability one. This follows directly by taking the limit as $t \to \infty, u \to \infty$. Theorem 4.1 can also be transformed to provide error bounds with respect to the best approximation, as given in the following.

Corollary 4.2. *Given the setting of Theorem 4.1, for all $u, t \geq 1$,*

$$\|\mathbf{A} - \mathbf{Q}\mathbf{Q}^T\mathbf{A}\|_F \leq \eta(r,p) \left(\sum_{k>r} \sigma_k(\boldsymbol{A})^2 \right)^{1/2} = \eta(r,p) \min_{rank(\boldsymbol{X}) \leq r} \|\boldsymbol{A} - \boldsymbol{X}\|_F \ ,$$

with

$$\eta(r,p) := 1 + t\sqrt{\frac{12r}{p}} + ut\frac{e\sqrt{r+p}}{p+1} \ ,$$

holds with probability at least $1 - 5t^{-p} - 2e^{-u^2/2}$.

Proof. The result is directly obtained from Theorem 4.1 and the Eckart–Young theorem (Theorem 3.5). □

4.2 Randomized HOSVD

In this section, the prototype algorithm of Section 4.1 is extended to a randomized higher order SVD for the Tucker format. This is possible in a rather straightforward way, by replacing the truncated matrix SVDs in the HOSVD algorithm (Algorithm 2) with randomized subspace projections $\boldsymbol{W}\boldsymbol{W}^T$ obtained by the prototype algorithm. This gives a Tucker representation with rank $(r_1 + p_1, \ldots, r_d + p_d)$, where r_μ and p_μ are the desired rank and oversampling parameter for the μ-th projection. To simplify the notation, we will use a uniform oversampling parameter $p_\mu = p$ in the following. The formal procedure is given in Algorithm 5.

Algorithm 5: Randomized Tucker subspace projection

```
Input  : Target $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, rank $(r_1, \ldots, r_d)$, oversampling $p \in \mathbb{N}$
Output : Core tensor $\mathcal{C} \in \mathbb{R}^{(r_1+p) \times \cdots \times (r_d+p)}$, basis matrices
         $\boldsymbol{W}_1 \in \mathbb{R}^{(r_1+p) \times n_1}, \ldots, \boldsymbol{W}_d \in \mathbb{R}^{(r_d+p) \times n_d}$
1 Set $\mathcal{X}_0 := \mathcal{X}$
2 for $\mu = 1, \ldots, d$ do
3     Create a Gaussian random matrix $\boldsymbol{G}_\mu \in \mathbb{R}^{(n_1 \cdot \ldots \cdot n_{\mu-1} \cdot n_{\mu+1} \cdot \ldots \cdot n_d) \times (r_\mu + p)}$
4     Calculate $\boldsymbol{A}_\mu := \boldsymbol{X}_{\mu-1}^{(\mu)} \boldsymbol{G}_\mu$
5     Calculate the QR-factorization $\boldsymbol{W}_\mu^T \boldsymbol{R}_\mu := \boldsymbol{A}_\mu$
6     Calculate $\mathcal{X}_\mu = \mathcal{X}_{\mu-1} \times_\mu \boldsymbol{W}_\mu^T$
7 Set $\mathcal{C} := \mathcal{X}_d$
```

Analogously to the matrix case, we can derive stochastic error bounds for this randomized HOSVD. To show this we require the following lemmas, which allow us to bound the effect on the singular values caused by μ-th mode contractions with (left-/right-)orthogonal matrices.

Lemma 4.3. *Let $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ be a tensor and let $\boldsymbol{A} \in \mathbb{R}^{n_\mu \times m}$ be a matrix. Then, for every $\nu \neq \mu$, the μ-th mode product with $\boldsymbol{A}$ can be represented in the ν-th mode matricization as*

$$\mathrm{Mat}_{(\nu)}\left(\mathcal{X} \times_\mu \boldsymbol{A}\right) = \boldsymbol{X}^{(\nu)} \tilde{\boldsymbol{A}} \ ,$$

with

$$\tilde{\boldsymbol{A}} := \mathrm{Mat}_{(1,3,\ldots,2d-3)}\left(\boldsymbol{I}_{n_1} \otimes \ldots \otimes \boldsymbol{I}_{n_{\mu-1}} \otimes \boldsymbol{A} \otimes \boldsymbol{I}_{n_{\mu+1}} \otimes \ldots \otimes \boldsymbol{I}_{n_{\nu-1}} \otimes \boldsymbol{I}_{n_{\nu+1}} \otimes \ldots \otimes \boldsymbol{I}_{n_d}\right)$$

if $\mu < \nu$ and

$$\tilde{\boldsymbol{A}} := \mathrm{Mat}_{(1,3,\ldots,2d-3)}\left(\boldsymbol{I}_{n_1} \otimes \ldots \otimes \boldsymbol{I}_{n_{\nu-1}} \otimes \boldsymbol{I}_{n_{\nu+1}} \otimes \ldots \otimes \boldsymbol{I}_{n_{\mu-1}} \otimes \boldsymbol{A} \otimes \boldsymbol{I}_{n_{\mu+1}} \otimes \ldots \otimes \boldsymbol{I}_{n_d}\right)$$

if $\mu > \nu$. Furthermore, if $\boldsymbol{A}$ is (left-/right-) orthogonal, then $\tilde{\boldsymbol{A}}$ is as well.

Proof. We show the result for $\mu < \nu$, as the case $\mu > \nu$ follows analogously. It holds that

$$\begin{aligned}
&\mathrm{Mat}_{(\nu)}\left(\mathcal{X} \times_\mu \boldsymbol{A}\right)\\
&= \mathrm{Mat}_{(\nu)}\left(\mathcal{X} \times_1 \boldsymbol{I}_{n_1} \ldots \times_{\mu-1} \boldsymbol{I}_{n_{\mu-1}} \times_\mu \boldsymbol{A} \times_{\mu+1} \boldsymbol{I}_{n_{\mu+1}} \ldots \times_{\nu-1} \boldsymbol{I}_{n_{\nu-1}} \times_{\nu+1} \boldsymbol{I}_{n_{\nu+1}} \ldots \times_d \boldsymbol{I}_{n_d}\right)\\
&= \mathrm{Mat}_{(\nu)}\left(\mathcal{X}\right) \mathrm{Mat}_{(1,3,..,2d-3)}\left(\boldsymbol{I}_{n_1} \otimes \ldots \otimes \boldsymbol{I}_{n_{\mu-1}} \otimes \boldsymbol{A} \otimes \boldsymbol{I}_{n_{\mu+1}} \otimes \ldots \otimes \boldsymbol{I}_{n_{\nu-1}} \otimes \boldsymbol{I}_{n_{\nu+1}} \otimes \ldots \otimes \boldsymbol{I}_{n_d}\right)\\
&= \boldsymbol{X}^{(\nu)} \tilde{\boldsymbol{A}} \ .
\end{aligned}$$

Now, if $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{I}$, then

$$\begin{aligned}
&\tilde{\boldsymbol{A}}\tilde{\boldsymbol{A}}^T\\
&= \mathrm{Mat}_{(1,3,\ldots,2d-3)}\Big(\boldsymbol{I}_{n_1}\boldsymbol{I}_{n_1}^T \otimes \ldots \otimes \boldsymbol{I}_{n_{\mu-1}}\boldsymbol{I}_{n_{\mu-1}}^T \otimes \boldsymbol{A}\boldsymbol{A}^T \otimes \boldsymbol{I}_{n_{\mu+1}}\boldsymbol{I}_{n_{\mu+1}}^T \otimes \ldots\\
&\quad \ldots \otimes \boldsymbol{I}_{n_{\nu-1}}\boldsymbol{I}_{n_{\nu-1}}^T \otimes \boldsymbol{I}_{n_{\nu+1}}\boldsymbol{I}_{n_{\nu+1}}^T \otimes \ldots \otimes \boldsymbol{I}_d\boldsymbol{I}_d^T\Big)\\
&= \mathrm{Mat}_{(1,3,\ldots,2d-3)}\left(\boldsymbol{I}_{n_1} \otimes \ldots \otimes \boldsymbol{I}_{n_{\mu-1}} \otimes \boldsymbol{I}_{n_\mu} \otimes \boldsymbol{I}_{n_{\mu+1}} \otimes \ldots \otimes \boldsymbol{I}_{n_{\nu-1}} \otimes \boldsymbol{I}_{n_{\nu+1}} \otimes \ldots \otimes \boldsymbol{I}_d\right)\\
&= \boldsymbol{I}_{n_1 n_2 \ldots n_{\nu-1} n_{\nu+1} \ldots n_d} \ .
\end{aligned}$$

Analogously, one can show that if $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}$ holds, then $\tilde{\boldsymbol{A}}^T\tilde{\boldsymbol{A}} = \boldsymbol{I}$ holds as well. □

Lemma 4.4. *For every tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ and every left-orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{n_\mu \times m}$ with $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}_m$, for the μ-th mode subspace projection*

$$\mathcal{Y} := \mathcal{X} \times_\mu \boldsymbol{Q}\boldsymbol{Q}^T \ ,$$

it holds that

$$\sigma_k\left(\boldsymbol{Y}^{(\nu)}\right) \leq \sigma_k\left(\boldsymbol{X}^{(\nu)}\right) \ ,$$

for all k.

Proof. In case $\mu = \nu$ we can write

$$\boldsymbol{Y}^{(\nu)} = \mathrm{Mat}_\nu\left(\mathcal{X} \times_\nu \boldsymbol{Q}\boldsymbol{Q}^T\right) = \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{X}^{(\nu)}$$

and the result follows directly from Propositions 3.8 and 3.7. For $\mu \neq \nu$, use the matrix representation $\tilde{\boldsymbol{Q}}$ from Lemma 4.3, i.e.

$$\boldsymbol{Y}^{(\nu)} = \mathrm{Mat}_\nu\left(\mathcal{X} \times_\mu \boldsymbol{Q}\boldsymbol{Q}^T\right) = \boldsymbol{X}^{(\nu)}\tilde{\boldsymbol{Q}}\tilde{\boldsymbol{Q}}^T \ .$$

Since $\boldsymbol{Q}$ is orthogonal, $\tilde{\boldsymbol{Q}}$ is orthogonal as well. Therefore, Propositions 3.8 and 3.7 are applicable and yield the desired result. □

Using these results, we can prove the following theorem, which quantifies the quality of this randomized Tucker subspace projection by providing a stochastic error bound.

Theorem 4.5 (Error Bound for the Randomized Tucker Subspace Projection)**.** *Let $\mathcal{X} \in \mathbb{R}^{n_1\times\ldots\times n_d}$ be a tensor, let $\boldsymbol{r}$ be a target rank and let $p \geq 4$ be an oversampling parameter. Then, for every $u, t \geq 1$, the randomized Tucker subspace projection given in Algorithm 5 fulfills*

$$\|\mathcal{X} - \mathcal{C} \times_1 \boldsymbol{W}_1 \ldots \times_d \boldsymbol{W}_d\|_F \leq \eta(r,p)\sqrt{d} \min_{T\text{-}rank(\mathcal{Y})\leq \boldsymbol{r}} \|\mathcal{X} - \mathcal{Y}\|_F \ ,$$

where $r = \max(r_1, \ldots, r_d)$, with probability at least $(1 - 5t^{-p} - 2e^{-u^2/2})^d$. The parameter η is given as

$$\eta(r,p) = 1 + t\sqrt{\frac{12r}{p}} + ut\frac{e\sqrt{r+p}}{p+1} \ .$$

Proof. By construction, $\mathcal{C}$ can be expressed in term of the $\boldsymbol{W}_\mu$ as $\mathcal{C} = \mathcal{X}\times_1\boldsymbol{W}_1^T \ldots \times_d\boldsymbol{W}_d^T$. Using $\mathcal{Z}_\mu = \mathcal{X} \times_1 \boldsymbol{W}_1^T\boldsymbol{W}_1 \ldots \times_\mu \boldsymbol{W}_\mu^T\boldsymbol{W}_\mu$, it follows that

$$\begin{aligned}
&\|\mathcal{X} - \mathcal{C} \times_1 \boldsymbol{W}_1 \ldots \times_d \boldsymbol{W}_d\|_F^2 \\
&= \left\|\mathcal{X} - \left(\mathcal{X} \times_1 \boldsymbol{W}_1^T \ldots \times_d \boldsymbol{W}_d^T\right) \times_1 \boldsymbol{W}_1 \ldots \times_d \boldsymbol{W}_d\right\|_F^2 \\
&= \left\|\mathcal{X} - \mathcal{X} \times_1 \boldsymbol{W}_1^T\boldsymbol{W}_1 \ldots \times_d \boldsymbol{W}_d^T\boldsymbol{W}_d\right\|_F^2 \\
&= \|\mathcal{X} - \mathcal{Z}_d\|_F^2 \\
&= \left\|\mathcal{Z}_0 \times_1 \left(\boldsymbol{I} - \boldsymbol{W}_1^T\boldsymbol{W}_1\right) + \mathcal{Z}_1 \times_2 \left(\boldsymbol{I} - \boldsymbol{W}_2^T\boldsymbol{W}_2\right) + \ldots + \mathcal{Z}_{d-1} \times_d \left(\boldsymbol{I} - \boldsymbol{W}_d^T\boldsymbol{W}_d\right)\right\|_F^2 \\
&= \left\|\mathcal{Z}_0 \times_1 \left(\boldsymbol{I} - \boldsymbol{W}_1^T\boldsymbol{W}_1\right)\right\|_F^2 + \left\|\mathcal{Z}_1 \times_2 \left(\boldsymbol{I} - \boldsymbol{W}_2^T\boldsymbol{W}_2\right)\right\|_F^2 + \ldots + \left\|\mathcal{Z}_{d-1} \times_d \left(\boldsymbol{I} - \boldsymbol{W}_d^T\boldsymbol{W}_d\right)\right\|_F^2 \\
&= \left\|\left(\boldsymbol{I} - \boldsymbol{W}_1\boldsymbol{W}_1^T\right)\boldsymbol{Z}_0^{(1)}\right\|_F^2 + \left\|\left(\boldsymbol{I} - \boldsymbol{W}_2\boldsymbol{W}_2^T\right)\boldsymbol{Z}_1^{(2)}\right\|_F^2 + \ldots + \left\|\left(\boldsymbol{I} - \boldsymbol{W}_d\boldsymbol{W}_d^T\right)\boldsymbol{Z}_{d-1}^{(d)}\right\|_F^2 \ ,
\end{aligned}$$

where we used the orthogonality of the summands. Repeated application of Lemma 4.4 provides that for all k and all μ

$$\sigma_k\left(\boldsymbol{Z}_{\mu-1}^{(\mu)}\right) = \sigma_k\left(\mathrm{Mat}_\mu\left(\mathcal{Z}_{\mu-2} \times_{\mu-1} \boldsymbol{W}_{\mu-1}^T\boldsymbol{W}_{\mu-1}\right)\right) \leq \sigma_k\left(\boldsymbol{Z}_{\mu-2}^{(\mu)}\right) \leq \sigma_k\left(\boldsymbol{X}^{(\mu)}\right)$$

holds. That is, in all involved matricizations, the singular values of the $\mathcal{Z}_\mu$ are bounded by the singular values of $\mathcal{X}$. For each μ, the algorithm obtains the orthogonal matrix $\boldsymbol{W}_\mu$ from a randomized matrix subspace approximation of $\boldsymbol{Z}_{\mu-1}^{(\mu)}$. Therefore, by applying Corollary 4.2,

$$\begin{aligned}\left\|\left(\boldsymbol{I}-\boldsymbol{W}_\mu\boldsymbol{W}_\mu^T\right)\boldsymbol{Z}_{\mu-1}^{(\mu)}\right\|_F &\leq \eta(r_\mu,p)\left(\sum_{k>r_\mu}\sigma_k^2\left(\boldsymbol{Z}_{\mu-1}^{(\mu)}\right)\right)^{1/2}\\ &\leq \eta(r_\mu,p)\left(\sum_{k>r_\mu}\sigma_k^2\left(\boldsymbol{X}^{(\mu)}\right)\right)^{1/2}\\ &\leq \eta(r,p)\min_{\text{T-rank}(\mathcal{Y})\leq\boldsymbol{r}}\|\mathcal{X}-\mathcal{Y}\|_F \qquad (4.2)\end{aligned}$$

holds with probability at least $1-5t^{-p}-2e^{-u^2/2}$ and $\eta(r,p)$ as defined in Corollary 4.2. As the random tensors are sampled independently for each μ, the probability that (4.2) holds simultaneously for all μ is at least $(1-5t^{-p}-2e^{-u^2/2})^d$, yielding the desired result. □

The randomized Tucker subspace projection obtained by Algorithm 5 yields a valid Tucker approximation of the input tensor with rank $(r_1+p,\ldots,r_d+p)$. This representation can be inexpensively truncated to the desired tucker rank $\boldsymbol{r}=(r_1,\ldots,r_d)$ using a deterministic HOSVD as described in Section 3.3.4.

4.3 Randomized TT-SVD

In this section, we show how the same idea of the randomized range approximation for matrices can be used to formulate a randomized algorithm that calculates an approximate TT-SVD of arbitrary tensors. This procedure is somewhat more involved than the one used for the Tucker format in the previous section. We show that stochastic error bounds analogous to the matrix and Tucker case can be obtained. Furthermore, we show that for sparse tensors this randomized TT-SVD can in fact be calculated in *linear* complexity with respect to the order of the tensor, instead of the exponential complexity of the deterministic TT-SVD.

The idea of our randomized TT-SVD procedure is to calculate nested range approximations, increasing by one mode at a time. The corresponding projectors are defined by left-orthogonal matrices $\boldsymbol{W}_\mu$, which are calculated using the randomized subspace projection given in Algorithm 6. This process is visualized in Figure 4.1. The dematricizations of these matrices will become the component tensors $\mathcal{W}_2,\ldots,\mathcal{W}_d$ of the final TT decomposition. The first component tensor $\mathcal{W}_1$ is given by contracting the initial $\mathcal{X}$ with all these orthogonal components (see Figure 4.2), i.e.

$$\mathcal{W}_1 := \mathcal{X} *_{(2,\ldots,d),(2,\ldots,d)} (\mathcal{W}_2\circ\ldots\circ\mathcal{W}_d) \ .$$

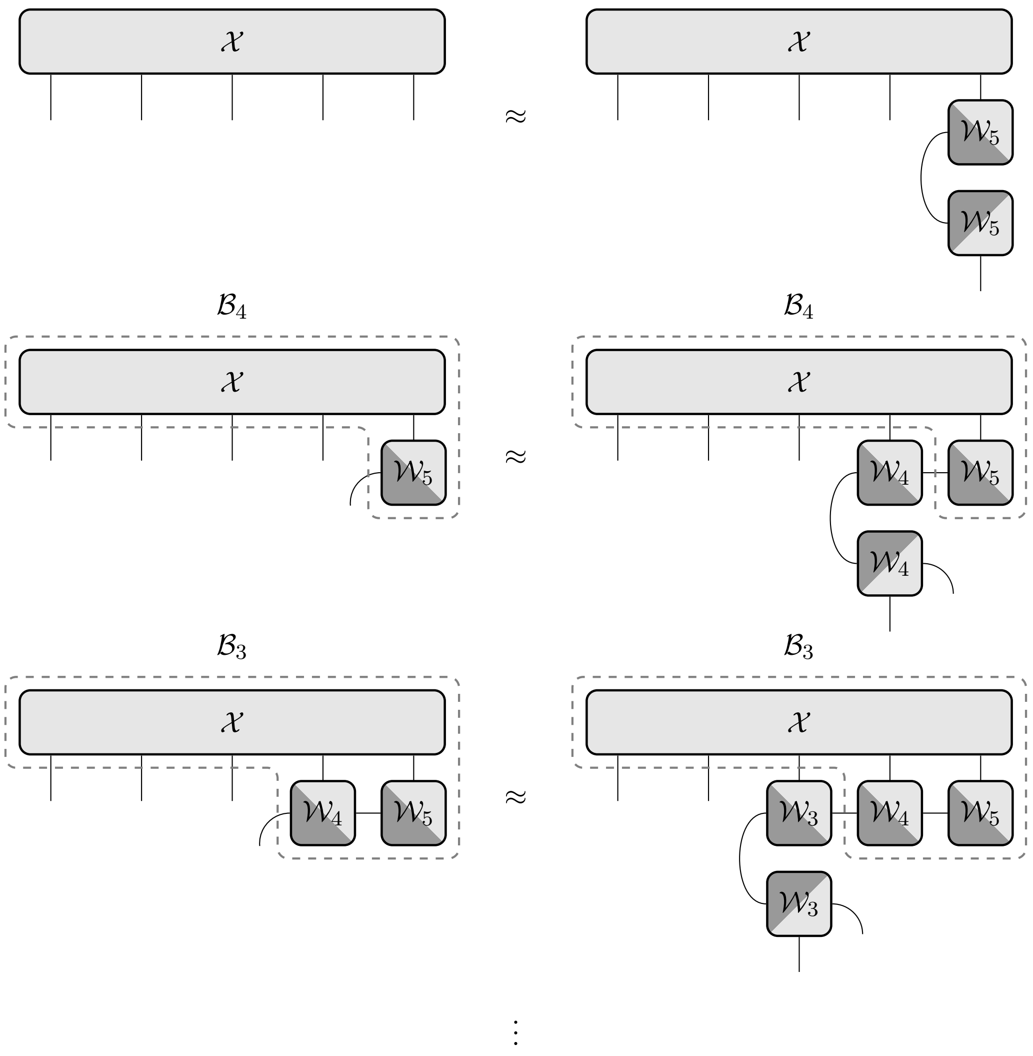


Figure 4.1: Iterative construction of the tensors $\mathcal{B}_\mu$ through subsequent range projections.

The complete procedure calculating the orthogonal components and this final contraction is given in Algorithm 6.

The tensor obtained as the result of Algorithm 6,

$$\mathcal{X} \approx \mathcal{W}_1 \circ \mathcal{W}_2 \circ \mathcal{W}_3 \circ \ldots \circ \mathcal{W}_d \ ,$$

is an approximate TT decomposition of rank $\boldsymbol{s} = (s_1, \ldots, s_{d-1})$. Recursively inserting the definition of the $\mathcal{B}_\mu$ from Algorithm 6 gives

$$\mathcal{B}_\mu = \mathcal{X} *_{(\mu+1,\ldots,d),(2,\ldots,d-\mu+1)} (\mathcal{W}_\mu \circ \ldots \circ \mathcal{W}_d) \ ,$$

and shows that the final composition can also be given in terms of contractions with the

```
Algorithm 6: Randomized TT Subspace Projection
Input    : Target $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, rank $(r_1, \dots, r_{d-1})$, oversampling $p \in \mathbb{N}$
Output : Component tensors $\mathcal{W}_1 \in \mathbb{R}^{n_1 \times r_1}, \mathcal{W}_2 \in \mathbb{R}^{r_1 \times n_2 \times r_2}, \dots, \mathcal{W}_d \in \mathbb{R}^{r_{d-1} \times n_d}$
1  Set $\mathcal{B}_d := \mathcal{X}$
2  for $\mu = d, \dots, 2$ do
3      Create a Gaussian random tensor $\mathcal{G}_\mu \in \mathbb{R}^{n_1 \times \dots \times n_{\mu-1} \times (r_{\mu-1}+p)}$
4      Calculate $\boldsymbol{A}_\mu^{<1>} := \left(\boldsymbol{G}_\mu^{<\mu-1>}\right)^T \boldsymbol{B}_\mu^{<\mu-1>}$
5      Calculate the RQ-factorization $\boldsymbol{R}_\mu \boldsymbol{W}_\mu^{<1>} := \boldsymbol{A}_\mu^{<1>}$
6      if $\mu = d$ then
7          Calculate $\mathcal{B}_{\mu-1} = \mathcal{B}_\mu *_{(\mu),(2)} \mathcal{W}_\mu$
8      else
9          Calculate $\mathcal{B}_{\mu-1} = \mathcal{B}_\mu *_{(\mu,\mu+1),(2,3)} \mathcal{W}_\mu$
10 Set $\mathcal{W}_1 = \mathcal{B}_1$
```

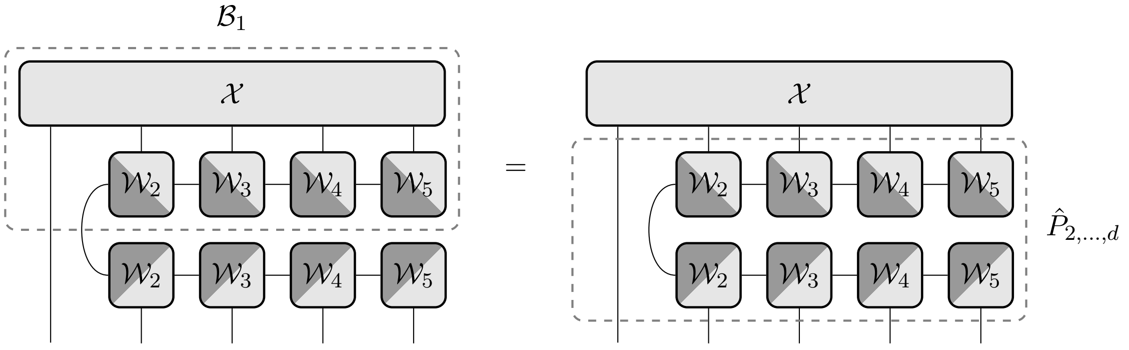


Figure 4.2: Depiction of the randomized TT-SVD as the action of the projection operator $\hat{P}_{2,\dots,d}$

orthogonal components $\mathcal{W}_\mu$, i.e.

$$\begin{aligned}
\mathcal{X} &\approx \mathcal{W}_1 \circ \mathcal{W}_2 \circ \mathcal{W}_3 \circ \dots \circ \mathcal{W}_d \\
&= \Big(\mathcal{X} *_{(2,\dots,d),(2,\dots,d)} (\mathcal{W}_2 \circ \mathcal{W}_3 \circ \dots \circ \mathcal{W}_d)\Big) \circ \mathcal{W}_2 \circ \mathcal{W}_3 \circ \dots \circ \mathcal{W}_d \\
&= \mathcal{X} \circ ((\mathcal{W}_2 \circ \mathcal{W}_3 \circ \dots \circ \mathcal{W}_d) *_{1,1} (\mathcal{W}_2 \circ \mathcal{W}_3 \circ \dots \circ \mathcal{W}_d)) \\
&=: \hat{P}_{2,\dots,d}(\mathcal{X})
\end{aligned}$$

This contraction can also be seen as the action of a projector $\hat{P}_{2,\dots,d}$, which is indeed an orthogonal projector, as the involved $\mathcal{W}_\mu$ are right-orthogonal. This relation is visualized in Figure 4.2. The following theorem shows that there exists a stochastic error bound for this randomized TT-SVD.

Theorem 4.6 (Error bound)**.** *Let $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ be a tensor, let $\boldsymbol{r}$ be a target rank and let $p \geq 4$ be an oversampling parameter. Then, for every $u, t \geq 1$, the error of the randomized*

TT-SVD as given in Algorithm 6 fulfills

$$\left\|\mathcal{X} - \hat{P}_{2..d}(\mathcal{X})\right\|_F \leq \sqrt{d-1}\,\eta(r,p) \min_{\text{TT-rank}(\mathcal{Y})\leq \boldsymbol{r}} \|\mathcal{X} - \mathcal{Y}\|_F \ ,$$

where $r = \max(r_1, \dots, r_{d-1})$, *with probability at least* $(1-5t^{-p}-2e^{-u^2/2})^{d-1}$. *The parameter* η *is given as*

$$\eta(r,p) = 1 + t\sqrt{\frac{12r}{p}} + ut\frac{e\sqrt{r+p}}{p+1} \ .$$

Proof. For notational convenience, let us define the right-orthogonal matrices $\boldsymbol{Q}_\mu := \boldsymbol{W}_\mu^{<1>}$. Since $\hat{P}_{2,\dots,d}$ is an orthogonal projector, we have

$$\begin{aligned}
\left\|\mathcal{X} - \hat{P}_{2,\dots,d}(\mathcal{X})\right\|_F^2 &= \|\mathcal{X}\|_F^2 - \left\|\hat{P}_{2..d}(\mathcal{X})\right\|_F^2 \\
&= \|\mathcal{X}\|_F^2 - \left\|\mathcal{X} *_{(2,\dots,d),(2,\dots,d)} (\mathcal{W}_2 \circ \ldots \circ \mathcal{W}_d) \circ (\mathcal{W}_2 \circ \ldots \circ \mathcal{W}_d)\right\|_F^2 \\
&= \|\mathcal{X}\|_F^2 - \left\|\mathcal{X} *_{(2,\dots,d),(2,\dots,d)} (\mathcal{W}_2 \circ \ldots \circ \mathcal{W}_d)\right\|_F^2 \\
&= \|\mathcal{X}\|_F^2 - \|\mathcal{B}_1\|_F^2 \\
&= \|\mathcal{X}\|_F^2 - \left\|\boldsymbol{B}_1^{<1>}\right\|_F^2 \ .
\end{aligned} \tag{4.3}$$

For all $1 \leq \mu < d$, it holds that

$$\begin{aligned}
\left\|\boldsymbol{B}_\mu^{<\mu>}\right\|_F^2 &= \left\langle \boldsymbol{B}_{\mu+1}^{<\mu>}\boldsymbol{Q}_\mu^T,\ \boldsymbol{B}_{\mu+1}^{<\mu>}\boldsymbol{Q}_\mu^T \right\rangle \\
&= \left\langle \boldsymbol{B}_{\mu+1}^{<\mu>},\ \boldsymbol{B}_{\mu+1}^{<\mu>}\boldsymbol{Q}_\mu^T\boldsymbol{Q}_\mu \right\rangle \\
&= \left\langle \boldsymbol{B}_{\mu+1}^{<\mu>},\ \boldsymbol{B}_{\mu+1}^{<\mu>} - \boldsymbol{B}_{\mu+1}^{<\mu>}(\boldsymbol{I} - \boldsymbol{Q}_\mu^T\boldsymbol{Q}_\mu) \right\rangle \\
&= \left\|\boldsymbol{B}_{\mu+1}^{<\mu>}\right\|_F^2 - \left\langle \boldsymbol{B}_{\mu+1}^{<\mu>}, \boldsymbol{B}_{\mu+1}^{<\mu>}(\boldsymbol{I} - \boldsymbol{Q}_\mu^T\boldsymbol{Q}_\mu) \right\rangle \\
&= \left\|\boldsymbol{B}_{\mu+1}^{<\mu+1>}\right\|_F^2 - \left\|\boldsymbol{B}_{\mu+1}^{<\mu>}(\boldsymbol{I} - \boldsymbol{Q}_\mu^T\boldsymbol{Q}_\mu)\right\|_F^2 \ ,
\end{aligned}$$

where we used that $\boldsymbol{I} - \boldsymbol{Q}_\mu^T\boldsymbol{Q}_\mu$ is an orthogonal projection as well. Inserting this iteratively into (4.3) gives

$$\begin{aligned}
\|\mathcal{X} - P_{2..d}(\mathcal{X})\|_F^2 &= \|\mathcal{X}\|_F^2 + \sum_{\mu=2}^{d} \left\|\boldsymbol{B}_{\mu+1}^{<\mu>}(\boldsymbol{I} - \boldsymbol{Q}_\mu^T\boldsymbol{Q}_\mu)\right\|_F^2 - \left\|\boldsymbol{B}_d^{<d>}\right\|_F^2 \\
&= \sum_{\mu=2}^{d} \left\|\boldsymbol{B}_{\mu+1}^{<\mu>}(\boldsymbol{I} - \boldsymbol{Q}_\mu^T\boldsymbol{Q}_\mu)\right\|_F^2 \ ,
\end{aligned}$$

where we used that $\boldsymbol{B}_d^{<d>} = \boldsymbol{X}^{<d>}$, as a matricization of $\mathcal{X}$ has the same norm as $\mathcal{X}$ itself. As each $\boldsymbol{Q}_\mu := \boldsymbol{W}_\mu^{<1>}$ is obtained in the exact setting of Theorem 4.1, we know that for all $2 \leq \mu \leq d$,

$$\begin{aligned}
&\left\| \boldsymbol{B}_{\mu+1}^{<\mu>} (\boldsymbol{I} - \boldsymbol{Q}_\mu^T \boldsymbol{Q}_\mu) \right\|_F^2 \\
&\leq \left[\left(1 + t\sqrt{\frac{12r}{p}}\right) \left(\sum_{k>r_\mu} \sigma_k^2(\boldsymbol{B}_{\mu+1}^{<\mu>}) \right)^{1/2} + ut\frac{e\sqrt{r+p}}{p+1} \sigma_{r_\mu+1}(\boldsymbol{B}_{\mu+1}^{<\mu>}) \right]^2 \\
&\leq \left[\left(1 + t\sqrt{\frac{12r}{p}} + ut\frac{e\sqrt{r+p}}{p+1}\right) \left(\sum_{k>r_\mu} \sigma_k^2(\boldsymbol{B}_{\mu+1}^{<\mu>}) \right)^{1/2} \right]^2 \\
&\leq \eta^2 \sum_{k>r_\mu} \sigma_k^2(\boldsymbol{B}_{\mu+1}^{<\mu>})
\end{aligned}$$

holds with probability at least $1 - 5t^{-p} - 2e^{-u^2/2}$. From the definition of the $\mathcal{B}_\mu$, it follows that the matricization can be expressed as

$$\boldsymbol{B}_{\mu+1}^{<\mu>} = \boldsymbol{X}^{<\mu>} (\mathcal{W}_{\mu+1} \circ \ldots \circ \mathcal{W}_d)^{<1>T} ,$$

i.e. as the matrix product of $\boldsymbol{X}^{<\mu>}$ and a left-orthogonal matrix. Using Proposition 3.8, it follows that the singular values of $\boldsymbol{B}_{\mu+1}^{<\mu>}$ are entrywise smaller than the ones of $\boldsymbol{X}^{<\mu>}$. Thereby, it follows that

$$\begin{aligned}
\left\| \boldsymbol{B}_{\mu+1}^{<\mu>} (\boldsymbol{I} - \boldsymbol{Q}_\mu^T \boldsymbol{Q}_\mu) \right\|_F^2 &\leq \eta^2 \sum_{k>r_\mu} \sigma_k^2(\boldsymbol{B}_{\mu+1}^{<\mu>}) \\
&\leq \eta^2 \sum_{k>r_\mu} \sigma_k^2(\boldsymbol{X}^{<\mu>}) \\
&\leq \eta^2 \min_{\operatorname{rank}(\boldsymbol{Y}^{<\mu>}) \leq r_\mu} \|\mathcal{X} - \mathcal{Y}\|_F^2 \\
&\leq \eta^2 \min_{\text{TT-rank}(\mathcal{Y}) \leq \boldsymbol{r}} \|\mathcal{X} - \mathcal{Y}\|_F^2 .
\end{aligned}$$

As the random matrices $\boldsymbol{G}$ in Algorithm 6 are sampled independently in each step, the combined probability that the above holds for all μ is at least $(1 - 5t^{-p} - 2e^{-u^2/2})^{d-1}$, as asserted. $\square$

Note that if the tensor $\mathcal{X}$ actually has TT-rank $\boldsymbol{r}$ or smaller, i.e. if $\min_{\text{TT-rank}(\mathcal{Y}) \leq \boldsymbol{r}} \|\mathcal{X} - \mathcal{Y}\| = 0$, then the randomized TT-SVD is exact with probability one. This follows directly from Theorem 4.6 by taking the limit $t \to \infty, u \to \infty$.

Using standard Gaussian random matrices $\boldsymbol{G}$, the computational complexity of the randomized TT-SVD is bounded by $\mathcal{O}(dsn^d)$, which is very similar to the deterministic TT-SVD presented in Section 3.4. However, as we show in the following proposition, for sparse tensors the complexity scales only *linearly* in the order, which is a dramatic reduction compared to the exponential scaling of the deterministic TT-SVD.

Proposition 4.7. *Assume that $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ contains at most N non-zero entries. Then the computational complexity of the randomized TT-SVD given in Algorithm 6 scales as $\mathcal{O}\big(d(r+p)^2(N + n(r+p))\big)$.*

Proof. First note that if $\mathcal{X}$ has at most N non-zero entries, then each $\mathcal{B}_\mu$ has at most $N(r_\mu + p)$ non-zero entries. The fact that $\mathcal{X}$ has at most N non-zero entries implies that, independent of μ, there are at most N tuples $(k_1, \dots, k_\mu)$, such that the subtensor $\mathcal{X}[k_1, \dots, k_\mu, \cdot, \dots, \cdot]$ is not completely zero. Now, each $\mathcal{B}_\mu$ can be given as $\mathcal{B}_\mu = \mathcal{X} *_{(\mu+1,\dots,d),(2,\dots,d-\mu+1)} (\mathcal{W}_\mu \circ \dots \circ \mathcal{W}_d)$. As any contraction involving a zero tensor results in a zero tensor, $\mathcal{B}_\mu$ is non-zero only if the first μ modes take values according to the aforementioned at most N tuples. As there is only one further mode of dimension $r_\mu + p$, there can in total be only $(r_\mu + p)N$ non-zero entries in $\mathcal{B}_\mu$.

There are at most $(r_{\mu-1} + p)(r_\mu + p)N$ entries of $\boldsymbol{G}$ actually needed to perform the (sparse) matrix product $\boldsymbol{A}_\mu^{<1>} := \left(\boldsymbol{G}_\mu^{<\mu-1>}\right)^T \boldsymbol{B}_\mu^{<\mu-1>}$ from Algorithm 6. Therefore, the complexity of this product can be bounded by $\mathcal{O}((r_{\mu-1} + p)(r_\mu + p)N)$. Calculating the *RQ*-factorization of $\boldsymbol{A}_\mu^{<1>}$ has complexity $\mathcal{O}((r_{\mu-1} + p)^2 n_\mu (r_\mu + p))$. The involved (de-)matricizations do not incur any computational costs. Finally, the product $\mathcal{B}_{\mu-1} = \mathcal{B}_\mu *_{(\mu,\mu+1),(2,3)} \mathcal{W}_\mu$ has computational complexity $\mathcal{O}(s_{j-1}s_jN)$. These steps are repeated $d-1$ times. The asymptotic cost is therefore bounded by $\mathcal{O}(d(N(r+p)^2 + n(r+p)^3))$, where $r := \max(r_1, \dots, r_{d-1})$. □

4.4 Structured Randomness and Relation to the ALS Algorithm

This section provides a brief outlook on the use of structured random tensors $\mathcal{G}_\mu$. To avoid repetition, we focus on the more interesting TT-SVD, but everything said straightforwardly extends to the randomized HOSVD as well. One possible advantage of the use of structured randomness is an improved computational complexity of the randomized HOSVD and TT-SVD. For the matrix case, the use of structured randomness is discussed in the work of Halko et al. [47], in particular the use of the subsampled random Fourier transform. Transferring their results to the high dimensional case allows a reduced computational complexity even if the given tensor is dense. However, as the computational complexity for matrices still scales as $\mathcal{O}(mn\log(r))$ (see [47, Section 4.6]), the exponential scaling for dense tensors remains. An even more interesting choice is therefore the use of random low rank tensors for $\mathcal{G}_\mu$. One easily verifies that if $\mathcal{G}_\mu$ is given in a TT representation of rank $\boldsymbol{q}$, the contraction in Line 4 of Algorithm 6 can be computed with decreased complexity depending on the representation of $\mathcal{X}$. In particular if $\mathcal{X}$ is given in a rank $\bar{\boldsymbol{q}}$ TT representation, then the complexity scales as $\mathcal{O}(\mu n q \bar{q} \min(q, \bar{q}) + n s_{\mu-1} s_\mu \min(q, \bar{q}))$. If $\mathcal{X}$ is sparse with at most N non-zero entries, then the complexity scales $\mathcal{O}(N\mu q^2 s_{\mu-1})$. This is of particular interest: if $\mathcal{X}$ is the sum of a sparse and a low rank tensor, the contraction can be performed separately for the two parts, due to the linearity of contractions. This results in an overall computational complexity that is the sum of the two aforementioned complexities, which in particular is linear in the order d. Using the arguments of Proposition 4.7, one shows that in this case all $\mathcal{B}_\mu$ can be represented as a sum of a sparse and a low rank tensor and therefore, the computational complexity of the complete algorithm scales only quadratically in the order (the outer iteration contributes one factor d). As mentioned in the introduction of this chapter, being able to compute the TT-SVD of such a sum of a sparse and a low rank tensor is at the very heart of several thresholding algorithms, most prominently the *iterative hard thresholding (IHT)* algorithm for tensor completion. Proving (stochastic) error bounds similar to Theorem 4.6 for the use of random low rank tensor is ongoing work and subject to future publications. However, the next section provides numerical evidence that such bounds indeed hold.

Apart from the outlined practical use as a means to approximate structured or sparse tensors, there is also a very noteworthy relation of the randomized TT-SVD and the popular *alternating least squares* (ALS) algorithm. The ALS itself is a general optimization algorithm, closely related to the very successful *DMRG* algorithm, which is well known in quantum physics. The following provides a minimal introduction of the algorithm, however, for an exhaustive treatment the reader is referred to the literature, e.g. the work of Holtz et al. [97], as well as Espig et al. [98].

The ALS is used to solve optimization problems with general objective functionals $\mathcal{J} : \mathbb{R}^{n_1\times\ldots\times n_d} \to \mathbb{R}$ on the low rank manifold $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ of tensors with fixed TT-rank $\boldsymbol{r}$. The special case of particular interest for this work is the approximation problem, i.e.

$$\mathcal{J}(\mathcal{X}) := \|\mathcal{Z} - \mathcal{X}\|_F$$

for a given tensor $\mathcal{Z} \in \mathbb{R}^{n_1\times\ldots\times n_d}$. The global optimum $\mathcal{X}^*$ is then the best rank $\boldsymbol{r}$ approximation

$$\mathcal{X}^* := \underset{\text{TT-rank}(\mathcal{X})=\boldsymbol{r}}{\operatorname{argmin}} \|\mathcal{Z} - \mathcal{X}\|_F \,.$$

The tensor train parameterization $\mathcal{X} = \tau(\mathcal{U}_1,\ldots,\mathcal{U}_d) = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d$ is multilinear in the parameters $\mathcal{U}_\mu \in \mathbb{R}^{r_{\mu-1}\times n_\mu\times r_\mu}$. Hence, fixing all components $\mathcal{U}_\mu$ except the ν-th component $\mathcal{U}_\nu$ provides a parameterization of $\mathcal{X}$ which is linear in the remaining component:

$$\mathcal{X} := \mathcal{X}(\mathcal{W}_\nu) := \mathcal{U}_1 \circ \ldots \circ \mathcal{W}_\nu \circ \ldots \circ \mathcal{U}_d \qquad\qquad \mathcal{W}_\nu \in \mathbb{R}^{r_{\nu-1}\times n_\nu\times r_\nu}$$

Now, the original optimization problem in the large ambient space $\mathbb{R}^{n_1\times\ldots\times n_d}$ induces an optimization problem

$$\mathcal{U}_\nu := \underset{\mathcal{W}_\nu\in\mathbb{R}^{r_{\nu-1}\times n_\nu\times r_\nu}}{\operatorname{argmin}} \|\mathcal{Z} - \mathcal{U}_1 \circ \ldots \circ \mathcal{W}_\nu \circ \ldots \circ \mathcal{U}_d\|_F = \underset{\mathcal{W}_\nu\in\mathbb{R}^{r_{\nu-1}\times n_\nu\times r_\nu}}{\operatorname{argmin}} \mathcal{J}(\mathcal{X}(\mathcal{W}_\nu))$$

in the relatively small subspace $\mathbb{R}^{r_{\nu-1}\times n_\nu\times r_\nu}$, which can be solved inexpensively. This procedure is continued iteratively by choosing another component to be optimized next, resulting in a nonlinear Gauß-Seidel iteration. From a practical point of view, it is favorable to optimize the components in the canonical order, i.e. from $\nu = 1$ to $\nu = d$, see e.g. [97]. The process of optimizing each component exactly once in this way is usually referred to as a half-sweep.

This idea of using a nonlinear Gauß-Seidel iteration is not limited to the tensor train format, but can also be applied to the Tucker, Hierarchical Tucker and even the canonical format. However, only the former two offer the possibility to keep the representation orthogonalized and only optimize at the core position. Doing so allows the ALS algorithm to perform much better and more stable than without orthogonalization, see e.g. [97]. To get started, the ALS algorithm requires an initial guess, i.e. it needs the $d-1$ components, which are fixed in the first step. A common choice is to use (Gaussian) random tensors for these, possibly orthogonalized. The interesting observation is that using this random initialization, one half-sweep is almost equivalent to the proposed randomized TT-SVD using random TT-tensors for $\mathcal{G}_\mu$, in the sense that numerically the same operations are performed. The only difference is that, from the point of view of the randomized TT-SVD, the random tensor $\mathcal{G}_\mu$ is not chosen as a Gaussian random tensor in each step, but as the first $d-\nu$ (contracted) random components of the ALS initialization. That is, $\mathcal{G}_\mu$ is a random TT-tensor as in the TT-SVD, but the component tensors are not changed during the iteration. Accordingly, note that for the matrix case $d=2$, the two methods actually coincide completely, as there is only one step. This makes an extension of the stochastic error bounds to the setting of possibly reused structured random tensors $\mathcal{G}_\mu$ very interesting, as such an extension would imminently provide stochastic error bounds for the first half-sweep of the ALS. Theoretically, this would be of major importance as there are mainly *local* convergence theories for the ALS, making the starting point essential.

4.5 Numerical Experiments

In order to provide practical evidence for the performance of the presented randomized TT-SVD, we conducted several numerical experiments. All calculations are performed using the `xerus` C++ tensor library (see Appendix A), which includes a complete implementation of the randomized TT-SVD. The experiments use several different types of random tensors, which are created as follows.

- **Standard Gaussian tensors** are created by definition by sampling each entry independently from $\mathcal{N}(0,1)$.
- **Sparse random tensors** are created by sampling N entries independently from $\mathcal{N}(0,1)$ and placing them at positions sampled independently and evenly distributed from $\mathbb{N}_{n_1} \times \mathbb{N}_{n_2} \times \ldots \times \mathbb{N}_{n_d}$.
- **Random rank $\boldsymbol{r}^*$ tensors** are created by sampling the entries of the components $\mathcal{U}_1, \ldots, \mathcal{U}_d$ in representation (3.12), independently from $\mathcal{N}(0,1)$, i.e. all components $\mathcal{U}_\mu$ are independent standard Gaussian random tensors.
- **Nearly rank $\boldsymbol{r}^*$ tensors** are created from a random rank $\boldsymbol{r}^*$ tensor $\tilde{\mathcal{X}} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ by adding a specific amount of Gaussian noise. That is, for a given noise level $\tau > 0$,

a standard Gaussian random tensor $\mathcal{G} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ is used to create the tensor

$$\mathcal{X} := \tilde{\mathcal{X}} + \tau \frac{\|\tilde{\mathcal{X}}\|_F}{\|\mathcal{G}\|_F} \mathcal{G} \ .$$

This yields a tensor which has a rank $\boldsymbol{r}^*$ approximation with relative error at most τ. For small ranks and large order the relative error of the best approximation indeed approaches τ.

- **Random tensors with roughly quadratically decaying singular values** are created from random low rank tensors by imposing the specific decay on the singular values of all relevant matricizations. To this end, for μ from 1 to $d-1$, the components $\mathcal{U}_\mu \circ \mathcal{U}_{\mu+1}$ are contracted and re-separated by calculating the SVD $\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{V}^T = \mathrm{Mat}_{(1,2)}(\mathcal{U}_\mu \circ \mathcal{U}_{\mu+1})$. The $\boldsymbol{\Sigma}$ factor is then replaced with a matrix $\tilde{\boldsymbol{\Sigma}}$, in which the singular values decay in the desired way. For a quadratic decay that is $\tilde{\boldsymbol{\Sigma}} := \mathrm{diag}(1, \frac{1}{2^2}, \frac{1}{3^2}, \dots, \frac{1}{100^2}, 0, \dots, 0)$, where 100 is a cut-off used in all of the following experiments. Subsequently, $\boldsymbol{U}_\mu^{<2>} = \boldsymbol{W}$ and $\boldsymbol{U}_{\mu+1}^{<1>} = \tilde{\boldsymbol{\Sigma}}\boldsymbol{V}^T$ are replaced. Note that since the later steps change the singular values of the earlier matricization, the singular values of the resulting tensor do *not* obey the desired decay exactly. However, empirically we observe that this method yields a rough approximation suitable for our experiments. If a better approximation is needed, the above procedure can be iterated. Empirically, we observe that such an iteration quickly converges towards a tensor for which the singular values of all relevant matricizations obey the specified decay.

In all experiments, uniform dimensions $n_\mu = n$, target ranks $r_\mu = r$ and (approximate) ranks of the solution $r_\mu^* = r^*$ are used. Deviating from this rule, the ranks are always chosen within the limits of the dimensions of the corresponding matricizations. For example, for $d = 12, n = 3, r = 10$ the actual TT-rank would be $\boldsymbol{r} = (3, 9, 10, \dots, 10, 9, 3)$. If not explicitly stated otherwise, 10000 samples are created for each setup. For each sample, we examine the result of the deterministic TT-SVD $\mathcal{X}_{\mathrm{det}}$, as described in Section 3.4.2, and of the randomized TT-SVD $\mathcal{X}_{\mathrm{rnd}}$ from Algorithm 6, *including* the subsequent (deterministic) rank r truncation. In particular, the relative errors

$$\epsilon_{\mathrm{det}} := \frac{\|\mathcal{X}_{\mathrm{det}} - \mathcal{X}\|_F}{\|\mathcal{X}\|_F} \qquad \epsilon_{\mathrm{rnd}} := \frac{\|\mathcal{X}_{\mathrm{rnd}} - \mathcal{X}\|_F}{\|\mathcal{X}\|_F}$$

are of interest. In some experiments we are also interested in the result of the random subspace projection $\mathcal{X}_{\mathrm{proj}}$, excluding the deterministic rank truncation. The relative error for this rank $r + p$ tensor is given analogously as

$$\epsilon_{\mathrm{proj}} := \frac{\|\mathcal{X}_{\mathrm{proj}} - \mathcal{X}\|_F}{\|\mathcal{X}\|_F} \ .$$

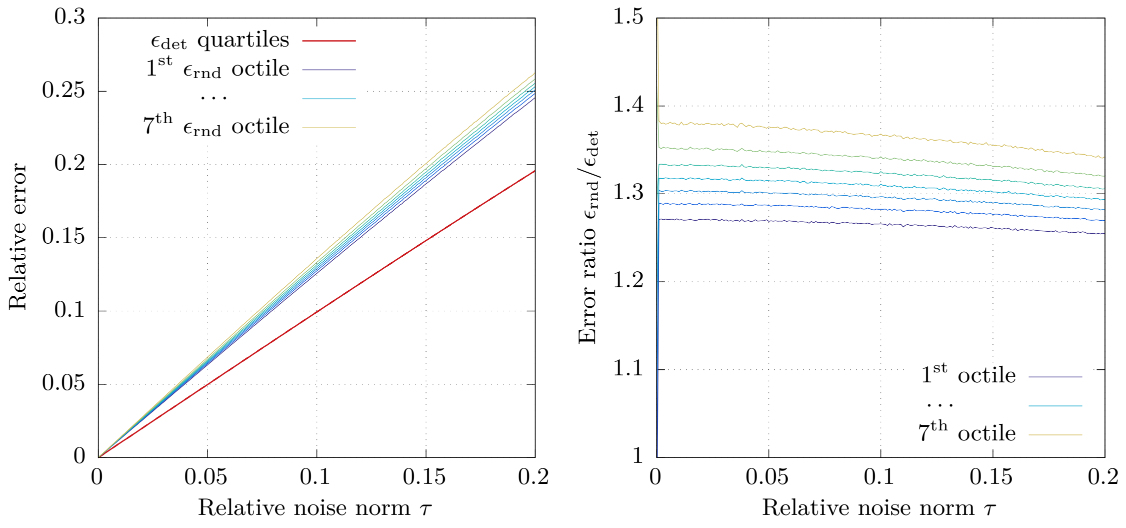


Figure 4.3: Approximation errors for the randomized and deterministic TT-SVD for nearly rank $r^* = 10$ tensors in dependence on the noise level τ. The target rank is chosen as $r = 10$ and the other parameters are $d = 12,\ n = 3, p = 10$.

4.5.1 Approximation Quality for Nearly Low Rank Tensors

In this first experiment, the approximation quality of the randomized TT-SVD for low rank tensors in dependence on noise is examined. To this end, nearly rank $\boldsymbol{r}^*$ tensors with variable noise levels τ are created and the randomized and deterministic TT-SVD are calculated for each sample. Figure 4.3 shows the resulting approximation errors for different noise levels τ. The other parameters are chosen as $d = 12, n = 3, r^* = 10, r = 10, p = 10$, with 10000 samples calculated for each noise level.

As expected, the error of the deterministic TT-SVD almost equals the noise τ for all samples, with no visible variance (the quartiles are indistinguishable). Almost independently of the noise level, the error ϵ_{rnd} of the randomized TT-SVD is on average larger by a factor of roughly 1.3. There is a very slight decay with increasing τ, meaning that the factor becomes smaller, as the target tensor approaches full rank. The only exception is in the case $\tau = 0$ where both methods are exact, up to machine precision. In contrast to the deterministic TT-SVD, there is some variance in the error ϵ_{rnd}. Relative to ϵ_{det}, the variance is almost constant (right plot in Figure 4.3), which means that the variance continuously decreases to zero with the noise level τ. Note that the error octiles are empirical error bounds for fixed probabilities $0.125, 0.25, 0.375, 0.5, 0.625, 0.75$ and 0.875. This allows a comparison to the theoretical expectations from Theorem 4.6, which postulates that for a fixed probability, the approximation error of the randomized TT-SVD is bounded by the error of the best approximation times a factor independent of noise. Indeed, each error octile in Figure 4.3 shows this expected behavior.

4.5.2 Approximation Quality with Respect to Oversampling

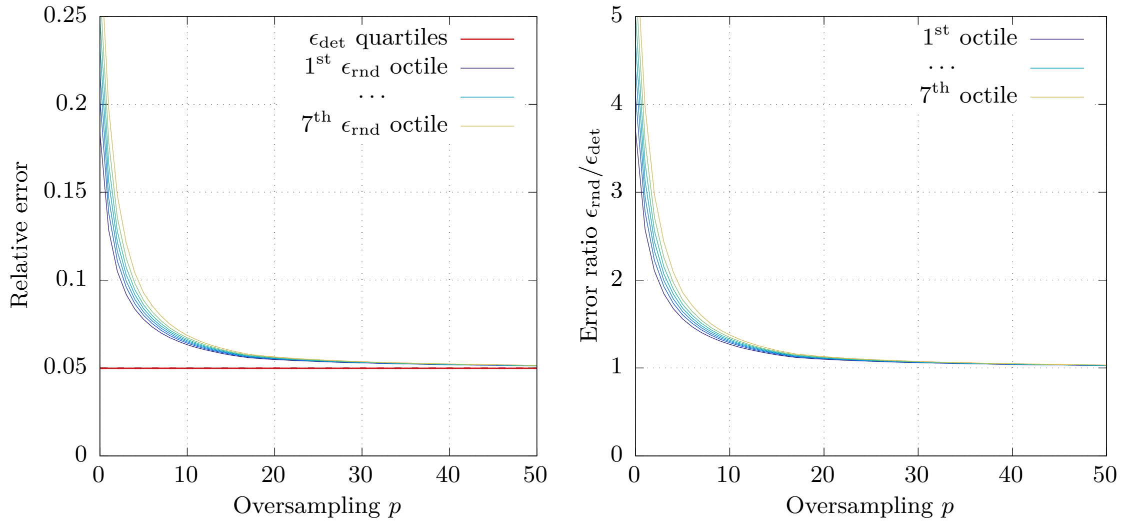


Figure 4.4: Approximation errors of the randomized and deterministic TT-SVD for nearly rank $r^* = 10$ tensors in dependence on the oversampling p. The target rank is chosen as $r = 10$ and the other parameters are $d = 12, n = 3, \tau = 0.05$.

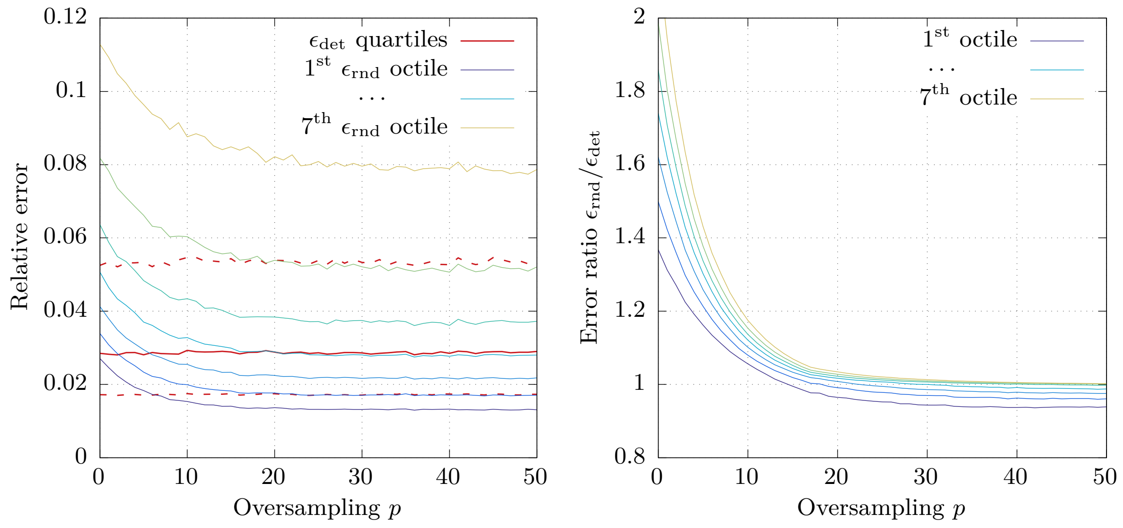


Figure 4.5: Approximation errors of the randomized and deterministic TT-SVD in dependence on the oversampling p for tensors with roughly quadratically decaying singular values. The target rank is chosen as $r = 10$ and the other parameters are $d = 12, n = 3$.

In this experiment, the influence of the oversampling parameter p on the approximation quality is examined. The first setting uses the same nearly low rank tensors as in Experiment 4.5.1, but now the noise level $\tau = 0.05$ is fixed and the oversampling parameter p is varied. Numerical results for the parameters $d = 12, n = 3, r^* = 10, r = 10, \tau = 0.05$ are shown in Figure 4.6. For small p, a steep descent of all error octiles is observed, which saturate towards an error of $\epsilon_{\mathrm{rnd}} = 0.05$, corresponding to the noise level and the deterministic error ϵ_{det}. With increasing p, the variance in ϵ_{rnd} approaches zero.

In the second setting, tensors with roughly quadratically decaying singular values are used. The numerical results are visualized in Figure 4.7 for the same parameters $d = 12, n = 3, r = 10$ as in the first setting. In contrast to the first setting, there is a significant variance in the error of the deterministic TT-SVD. This variance is in part explained by the fact that the deterministic TT-SVD is only a quasi-best approximation (see Theorem 3.22). Additionally, in this setting, the error of the best approximation varies significantly. The decay of the relative error of the randomized TT-SVD is observed as before, although it is less distinct. Examining the ratio $\epsilon_{\text{rnd}}/\epsilon_{\text{det}}$, we observe almost the same decay with increasing p as in the first setting. The most significant difference is that for large p, most octiles are actually smaller than one, i.e. in most cases, the error of the randomized TT-SVD is *smaller* than the one of the deterministic TT-SVD. Note that this is no contradiction, as the deterministic TT-SVD only gives a quasi-best approximation.

For a fixed probability, i.e. fixed u and t, Theorem 4.6 predicts a dependency on p of $c_1 + \frac{c_2}{\sqrt{p}} + \frac{c_3\sqrt{r+p}}{p+1}$ for both the error and the error relative to the best approximation. This is roughly what is observed in both experiments, with the exception that the decay for small p is even somewhat faster than $1/p$, i.e. the error decays faster than the upper error bound provided by Theorem 4.6.

4.5.3 Approximation Quality with Respect to the Order

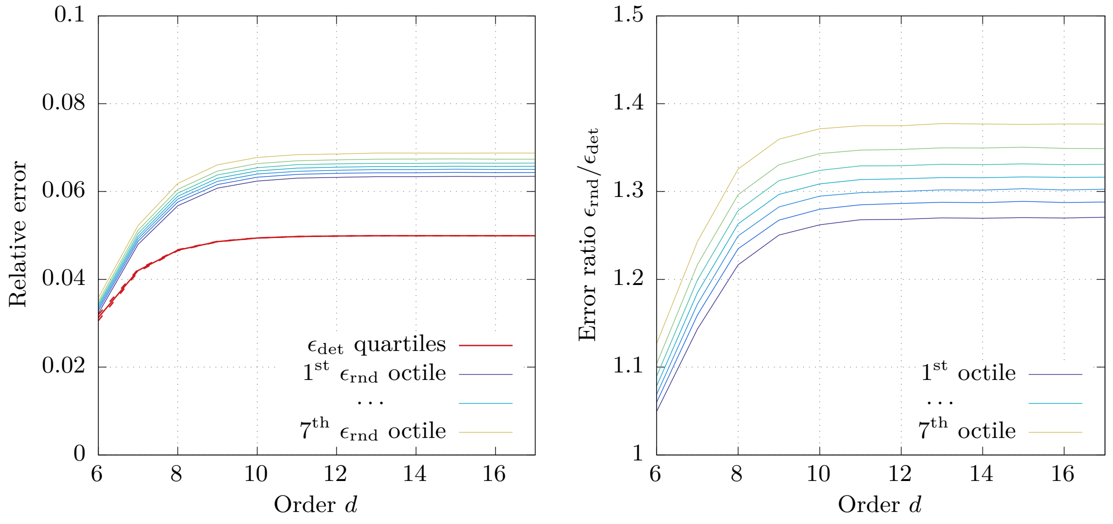


Figure 4.6: Approximation errors of the randomized and deterministic TT-SVD in dependence on the order for nearly rank $r^* = 10$ tensors. The parameters are $n = 3, r = 10, \tau = 0.05$.

In this experiment, the impact of the order on the approximation quality is investigated. Again, the first setting uses nearly low rank tensors with some noise. The parameters are chosen as $d = 12, n = 3, r^* = 10, r = 10, \tau = 0.05$. The results are shown in Figure 4.6. With these parameters, there is no truncation for $d < 6$, since all ranks are automatically smaller than 10. With increased order the maximal possible ranks increase, leading to an

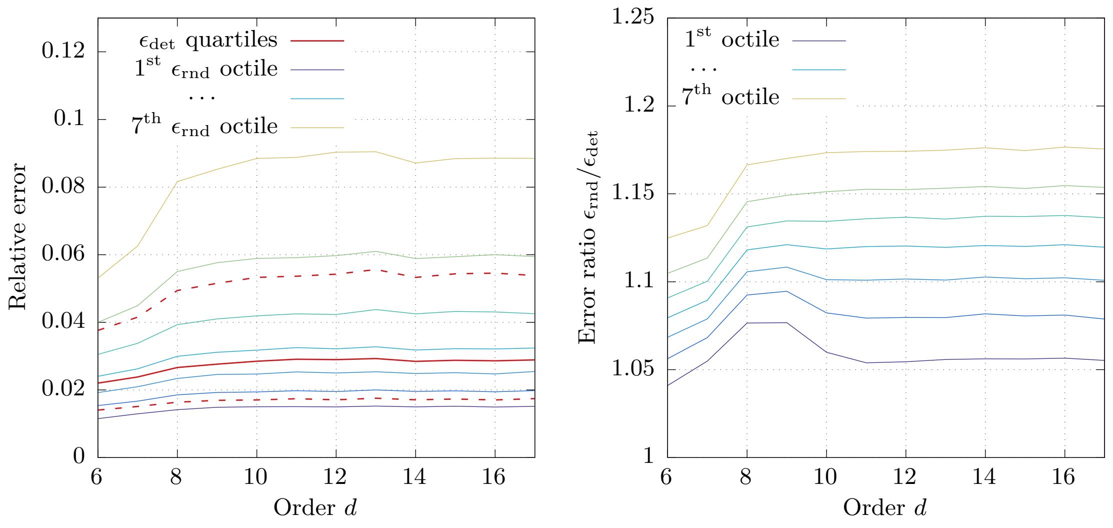


Figure 4.7: Approximation errors of the randomized and deterministic TT-SVD in dependence on the order for tensors with roughly quadratically decaying singular values. The parameters are $n = 3, r = 10$.

actual truncation/approximation of the tensor. This is visible in Figure 4.6, as the relative error of both SVDs, as well as the error ratio, increase with the order at the beginning. For larger d, both errors are almost constant and, therefore, also the ratio $\epsilon_{\text{rnd}}/\epsilon_{\text{det}}$, for which the median (4th octile) remains slightly larger than 1.3, independent of the order.

The second setting uses target tensors with roughly quadratically decaying singular values. The results for the otherwise same parameters $d = 12, n = 3, r = 10$ are shown in Figure 4.7. Again, both errors are almost independent of d, except for the effect of small orders. The major difference is, as was already observed in Experiment 4.5.2, that there is a significant variance in ϵ_{det} for these target tensors as well.

The observed results are somewhat better than expected from Theorem 4.6. In fact, the error bounds for both the deterministic and the randomized TT-SVD include a factor $\sqrt{d-1}$, which is not visible in the examined settings. For the randomized TT-SVD, the order also appears as an exponent in the probability, which should be observable in these results for the ratios $\epsilon_{\text{rnd}}/\epsilon_{\text{det}}$. The fact that it is not suggest that a refinement of Theorem 4.6 is possible, in which this exponent does not appear.

4.5.4 Computation Time

In this experiment, the computational complexity of the (randomized) TT-SVD is examined, in particular regarding the expected linear scaling with respect to the order for sparse tensors. To this end, random sparse tensors with varying order and a fixed number $N = 1000$ of entries are created and the computation time for the deterministic and randomized TT-SVD is measured. The other parameters are chosen as $n = 2, r = 10, p = 10$.

Figure 4.8 show the results, which clearly confirm the linear scaling of the randomized TT-SVD. The absolute runtimes are of course hardware and implementation dependent[1], however, the dramatic advantage of the randomized approach over the deterministic TT-SVD for high orders is obvious.

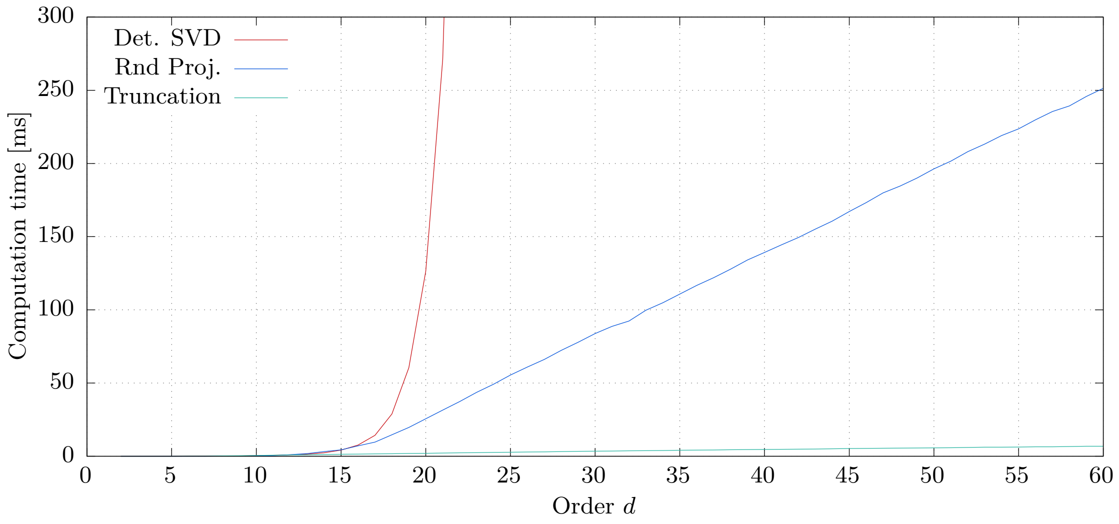


Figure 4.8: Average run-time of the det. and rnd. TT-SVD algorithms for sparse tensors of different orders. The time for the random subspace projection and the rank truncation of the rnd. TT-SVD is given separately. The parameters are $n = 2,\ r = 10,\ p = 10, N = 1000$.

4.5.5 Approximation Quality using Structured Randomness

This experiment examines the use of structured randomness as discussed in Section 4.4. To this end, variants of Algorithm 6 are used, which use structured random tensors $\mathcal{G}_\mu \in \mathbb{R}^{n_1\times\ldots\times n_{\mu-1}\times(r_{\mu-1}+p)}$. The first choice of structured random tensors are TT-rank $(r_{\mu-1}+p,\ldots,r_{\mu-1}+p)$ tensors[2]

$$\mathcal{G}_\mu = \mathcal{V}_1 \circ \mathcal{V}_2 \circ \ldots \circ \mathcal{V}_{\mu-1}\ ,$$

where $\mathcal{V}_1 \in \mathbb{R}^{n_1\times(r_{\mu-1}+p)}$ and $\mathcal{V}_\nu \in \mathbb{R}^{(r_{\mu-1}+p)\times n_\nu\times(r_{\mu-1}+p)}$ for $\nu = 2,\ldots,\mu-2$ and $\mathcal{V}_{\mu-1} \in \mathbb{R}^{(r_{\mu-1}+p)\times n_{\mu-1}\times(r_{\mu-1}+p)}$ are standard Gaussian tensors.[3] The second choice is the use of rank one tensors for each index position, that is

$$\mathcal{G}_\mu = \sum_{i=1}^{r_{\mu-1}+p} \boldsymbol{g}_{1,i}\otimes\boldsymbol{g}_{2,i}\otimes\ldots\otimes\boldsymbol{g}_{\mu-1,i}\otimes\hat{\boldsymbol{e}}_i\ ,$$

[1]The results in Figure 4.8 were achieved using the **xerus** C++ tensor library (see Appendix A) on a standard laptop with Intel Core i7-5600U CPU and 8 GB RAM

[2]The TT-rank is chosen within the limits possible, i.e. the first ranks might be smaller depending on the external dimensions.

[3]Since the last component $\mathcal{W}_{\mu-1}$ carries two external indices, this is not strictly a TT representation. However, by separating the last index into a further component, full rank component one obtains an equivalent strict TT-representation.

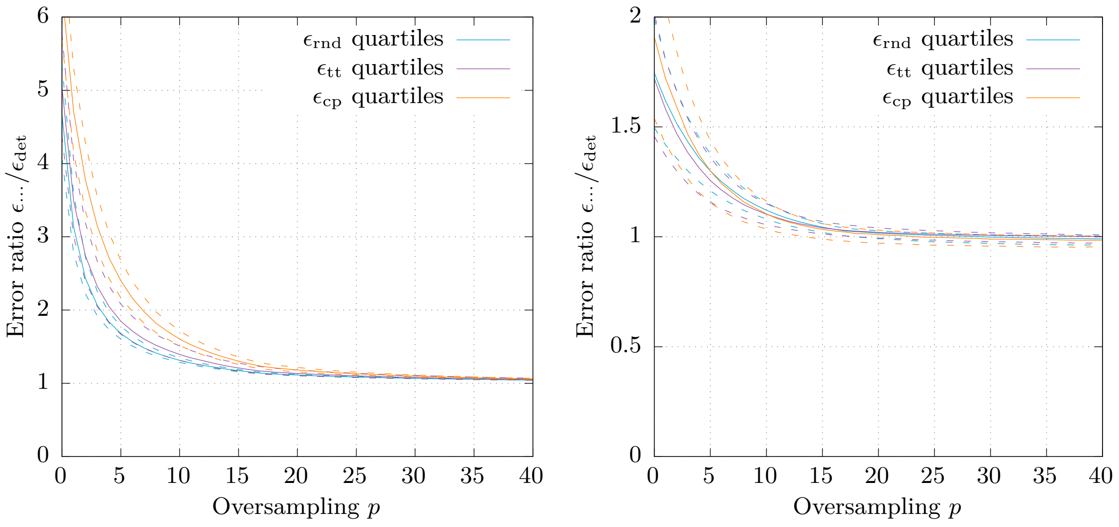


Figure 4.9: Approximation errors with respect to the deterministic TT-SVD of the randomized TT-SVD, using Gaussian random tensors (ϵ_{rnd}), random TT-tensors (ϵ_{tt}) and sums of random rank one tensors (ϵ_{cp}). The oversampling p is varied and the parameters are $d = 12, n = 3, r = 10$. On the left for nearly rank $r^* = 10$ tensors with noise $\tau = 0.05$, on the right for tensors with roughly quadratically decaying singular values.

where the $\boldsymbol{g}_{\nu,i} \in \mathbb{R}^{n_\nu}$ are standard Gaussian vectors and $\hat{\boldsymbol{e}}_i \in \mathbb{R}^{s_{\mu-1}}$ denotes the i-th standard basis vector. Figure 4.9 shows the reconstruction quality relative to the deterministic TT-SVD for nearly low rank tensors and tensors with roughly quadratically decaying singular values in dependence of the oversampling parameter p. Figure 4.10 uses the same settings and shows the relative reconstruction quality in dependence of the order.

The main observation is that the qualitative behavior does not differ between the different choices of randomness, with the notable exception that using rank one random tensors, the approximation quality for nearly low rank tensors significantly deteriorates with increased order, see Figure 4.10. The use of a dense random tensor shows mostly better results than the use of structured random tensors, although the difference is rather small in the examined settings. Especially in Figure 4.9, it is apparent that with slightly increased oversampling parameter, the TT-SVD with structured randomness achieves the same accuracy as the one using Gaussian random tensors.

4.5.6 Comparison to Alternating Least Squares

In this final experiment, the performance of the randomized TT-SVD is compared to randomly initialized ALS iterations. In all settings, the results of a single and of four ALS half-sweeps with random rank $r + p$ initialization and subsequent rank r truncation are used. Figure 4.11 shows the reconstruction quality relative to the deterministic TT-SVD for nearly low rank tensors and tensors with roughly quadratically decaying singular values in dependence on the oversampling parameter p. Figure 4.12 uses the same settings and

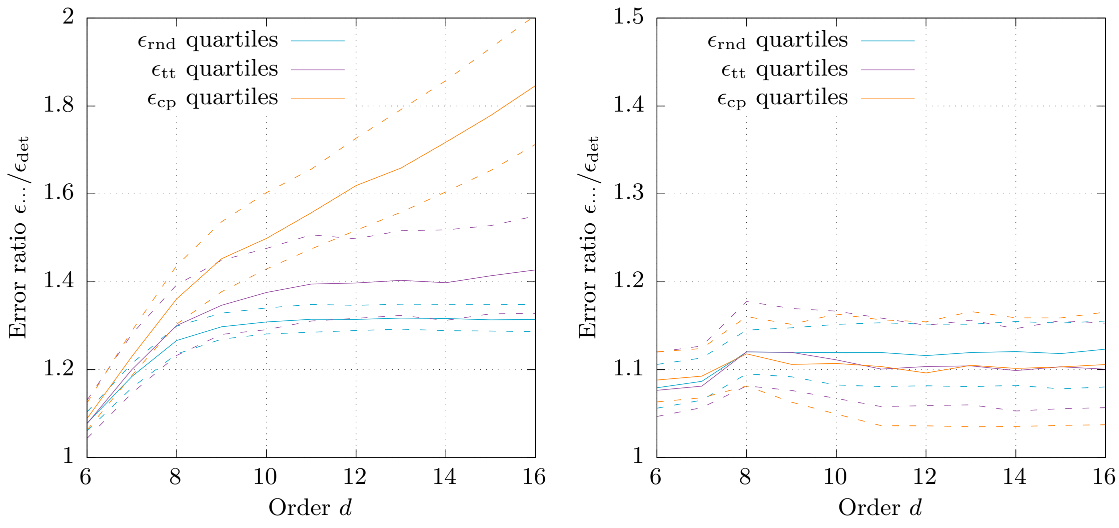


Figure 4.10: Approximation errors with respect to the deterministic TT-SVD of the randomized TT-SVD, using Gaussian random tensors (ϵ_{rnd}), random TT-tensors (ϵ_{tt}) and sums of random rank one tensors (ϵ_{cp}). The order d is varied and the parameters are $p = 10, n = 3, r = 10$. On the left for nearly rank $r^* = 10$ tensors with noise $\tau = 0.05$, on the right for tensors with roughly quadratically decaying singular values.

shows the relative reconstruction quality in dependence on the order. In both figures, the approximation quality of the randomized TT-SVD *without* the rank truncation, i.e. for the rank $r + p$ tensors, is given.

There are several noteworthy observations. The approximation quality of a single ALS half-sweep is roughly the same as that of the randomized TT-SVD. Without oversampling, this is significantly worse than the deterministic TT-SVD but not arbitrarily bad, e.g. much better than a random tensor. This is noteworthy, since from the point of view of the ALS, the result is only a single optimization step applied to a random initial guess. Increasing the number of half-sweeps to four yields approximations, which usually have slightly smaller error than the ones of the deterministic TT-SVD, independent of the oversampling.

The error obtained by the randomized subspace projection without the subsequent rank truncation is quite different for the two settings examined, see Figure 4.11. For nearly low rank tensors, the error is, independent of p, actually larger than the error of the complete randomized TT-SVD, despite the rank $\boldsymbol{r} + \boldsymbol{p}$ being larger. This means that in the nearly low rank setting, the rank truncation acts as some kind of denoising, removing some of the error introduced by the random projection. For tensors with roughly quadratically decaying singular values, the error of the projection keeps decreasing with increased p, which is expected, since here the higher rank allows a significantly better approximation.

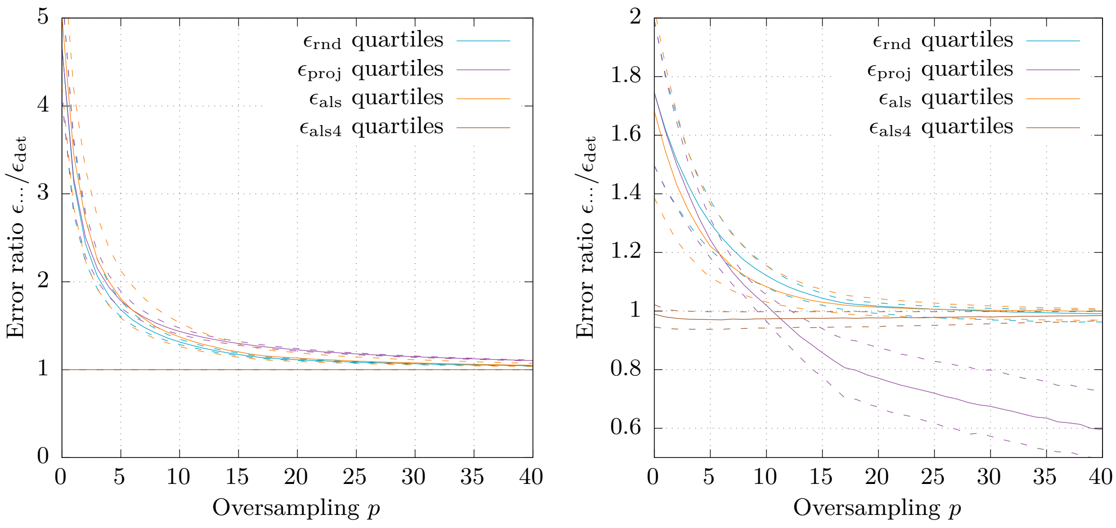


Figure 4.11: Approximation errors of a single and of four ALS half-sweeps with random rank $r+p$ initialization and subsequent rank r truncation, relative to the error of the deterministic TT-SVD. The oversampling p is varied and the parameters are $d = 12, n = 3, r = 10$. On the left for nearly rank $r^* = 10$ tensors with noise $\tau = 0.05$, on the right for tensors with roughly quadratically decaying singular values.

4.6 Summary and Conclusions

We have shown theoretically and practically that the introduced randomized HOSVD and TT-SVD algorithms provide a robust alternative to the deterministic algorithms, incurring only a controlled stochastic error. In particular, the randomized HOSVD and TT-SVD are exact if applied to a tensor with TT-rank smaller than or equal to the target rank. For the case of actual low rank approximations, stochastic error bounds were proven. A number of numerical experiments were conducted to confirm these bounds in practice. Some numerical experiments suggest that these proven bounds are even somewhat pessimistic, as the observed error is often significantly smaller than expected. In particular, no significant deterioration of the error bounds with increased order is observed, contrary to Theorem 4.6, which includes d as exponent in the probability. This leaves room for improvements and it is the author's expectation that enhanced versions of our theorem are possible. On the computational side, we have provided efficient implementations of the proposed algorithm, available in the `xerus` C++ library (see Appendix A). For sparse tensors, we have shown that the randomized TT-SVD algorithm dramatically outperforms the deterministic algorithm, scaling only linearly instead of exponentially in the order, which was verified by this implementation. We believe that these results show that the randomized TT-SVD algorithm is a useful tool for low rank approximations of higher order tensors.

Apart from enhanced error bounds, a topic of further investigations is the use of structured random tensors in the randomized TT-SVD. For the matrix case, several choices of struc-

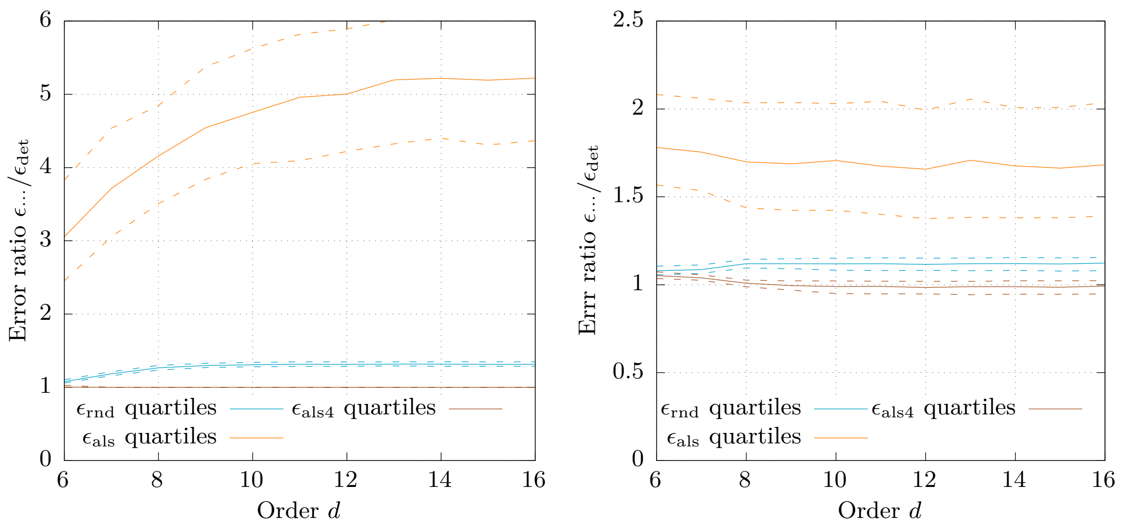


Figure 4.12: Approximation errors of a single and of four ALS half-sweeps with random rank r initialization relative to the error of the deterministic TT-SVD. The order d is varied and the parameters are $d = 12, n = 3, r = 10$. On the left for nearly rank $r^* = 10$ tensors with noise $\tau = 0.05$, on the right for tensors with roughly quadratically decaying singular values.

tured random matrices are already discussed in the work of Halko et al. [47]. Transferring their results to the high dimensional case could allow reduced computational cost even for dense tensors, as is the case for matrices. The even more interesting choice, however, is the use of random low rank tensors, as discussed in Section 4.4. On the one hand, an analysis of this setting would benefit the theoretical understanding of the alternating least squares algorithm, as it would result in error bound for the first half-sweep for a random initial guess. This can be of major importance, as there are mainly *local* convergence theories for the ALS so far, which is why the starting point is essential. On the other hand, obtaining error bounds for this setting would also allow computationally fast application of the randomized TT-SVD to tensors given in various data-sparse formats, e.g. in a sparse representation, the canonical, the TT or HT format and also sums of those. The latter could directly benefit several thresholding algorithms, for example the iterative hard thresholding algorithm for tensor completion, discussed in the next chapter.

5 Tensor Recovery

In the previous chapters, the deterministic and randomized HOSVD and TT-SVD were introduced, which allow the computation of exact and approximate Tucker or Tensor Train representations of a given tensor. The deterministic variants assume that the complete tensor is known and available for the calculation. This implies that they are affected by the curse of dimensionality, i.e. the exponential scaling with respect to the order, as they have to handle the complete tensor. For large orders this becomes prohibitively expensive. The randomized SVD variants introduced in the previous chapter do not require knowledge of the complete tensor, instead only requiring the ability to calculate contractions with test tensors. If the tensor is given in some structured way, this can lead to severely reduced computational cost, scaling sub-exponentially in the order, as was shown in the previous chapter. Still, the requirement to compute various contractions with (random) test tensors is a rather strong one and cost reductions are only possible if these calculations are cheap.

One possible remedy are so-called cross approximation algorithms, which only require the (adaptive) evaluation of single entries of the input tensor. For matrices it is well known that the knowledge of r linearly independent rows and columns is sufficient to reconstruct any rank r matrix and directly obtain a rank r factorization of said matrix. However, finding these linearly independent rows and columns might require more information than just those entries. Additionally, if the matrix is not (exactly) of rank r, the quality of the approximation depends on the chosen rows and columns. Here, one aims to find the $r \times r$ submatrix with maximum volume (i.e. absolute value of the determinant), see e.g. the work of Goreinov et al. [99]. Both problems can be tackled by adaptive cross approximation algorithms, which try to find “good” sub-matrices using as few evaluations as possible [100–102]. The idea of cross approximation can be extended to higher order tensors in the Tucker [103, 104], tensor train [86] and hierarchical Tucker format [105]. For the TT and HT format, these methods often work with linear complexity with respect to the order.

This chapter is concerned with methods to further relax these requirements and enable the reconstruction of low rank tensors from more general measurements. This in particular includes settings where the measurements are a priori given and no further/adaptive information is available at all, e.g. tensors describing physical systems, which are only available through limited and possibly completely predefined measurements. The general term for this endeavor is *tensor recovery*, as one attempts to recover a low rank representation of the tensor from incomplete information. The fundamental idea of tensor recovery goes back to compressive sensing, which is based on the postulate that if an object of interest can be defined completely by a specific number of parameters, then one should be able to recover said object from a similar amount of independent measurements. For tensor recovery, this means that if a tensor has e.g. a small TT-rank, then it should be possible to recover it from a number of independent measurements related to the number of degrees of freedom possessed by tensors of that rank.

This chapter starts with an introduction to the very popular and successful low rank matrix recovery setting, which builds the foundation for the multidimensional extension to higher order tensors introduced in Section 5.2. The algorithmic treatment of the tensor recovery problem is discussed in Section 5.3 and numerically verified in Section 5.4. Finally, Section 5.5 summarizes and discusses the results.

Parts of the introduction in Sections 5.1 and 5.2 were previously published in [106].

5.1 The case $d = 2$: Matrix Recovery

As a motivation and an important point of reference for the general tensor recovery setting, this section introduces the matrix recovery setting, i.e. the special case restricted to order two. The advantage of the matrix setting is that it is already well understood and more intuitive than the recovery of higher order tensors. As will be shown later, it is also special in the sense that there are unfortunately no straightforward generalizations for many of the results presented in this section. However, this makes the matrix results no less important for tensor recovery, as high dimensional extensions are still sought-after.

In *matrix recovery*, one aims to reconstruct an unknown matrix from limited knowledge about its entries, using the assumption that it has a low rank. An important special case is *matrix completion*, where an unknown matrix is to be reconstructed from knowledge of some of its entries. A famous and very instructive example for a matrix completion problem is the so-called *Netflix Problem*, described in detail by Bennett, Lanning, et al. [107]. In slightly simplified form, it can be introduced as follows. Imagine a large video lending platform, with numerous customers and available videos. The company collects ratings from their customers whenever these watch one of the videos. These ratings can be organized as a matrix, with each column representing a specific user and each row a specific video title. Over time, the company obtains a large number of entries of this rating matrix. However, the vast majority of entries is unknown, since every user usually only rates a tiny fraction of the available videos. The company wants to use this data to recommend videos to its costumers, in such a way that the user is likely to like the recommended video, i.e. give it a good rating. In other words, they want to know *all* entries of the rating matrix and thereby not only know what ratings a user gave to the videos they have watched, but also what rating they would give a video they have not watched yet. Obtaining such predictions works quite well as summarized by Bell and Koren [108], who won the “Netflix Prize”, offered by Netflix for an efficient algorithm to solve this problem. The key insight is, that the preferences of people are not completely random, but rather follow some potentially unknown but existing patterns. In the context of matrix completion, this insight justifies the conjecture that the complete rating matrix is approximately of low rank. As described in the remaining section, this low rank assumption often allows a matrix to be reconstructed from a highly incomplete set of entries. Obviously, matrix completion and matrix recovery are not limited to customer recommendation systems. Successful applications include machine learning [109–111], computer vision [112], global positioning [113, 114], quantum state tomography [12] and many more.

5.1.1 Formal Definition

This section provides formal definitions for the matrix recovery setting and its special case, the matrix completion setting. For this, let $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ be an unknown matrix that shall be reconstructed. Assume that N linear measurements

$$\boldsymbol{b}[k] := \sum_{i=1}^{m}\sum_{j=1}^{n} \mathcal{A}_{k,i,j}\ \boldsymbol{M}[i,j] \qquad\qquad \mathcal{A}_{k,i,j} \in \mathbb{R}, \quad k = 1,\dots,N$$

of $\boldsymbol{M}$ are known. It is convenient to introduce the linear measurement operator $\hat{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^N$, defined entrywise as

$$\hat{\mathcal{A}}(\boldsymbol{X})[k] := \sum_{i=1}^{m}\sum_{j=1}^{n} \mathcal{A}_{k,i,j}\ \boldsymbol{X}[i,j]$$

and its adjoint operator $\hat{\mathcal{A}}^T : \mathbb{R}^N \to \mathbb{R}^{m\times n}$, defined entrywise as

$$\hat{\mathcal{A}}^T(\boldsymbol{v})[i,j] := \sum_{k=1}^{N} \mathcal{A}_{k,i,j}\ \boldsymbol{v}[k] \ .$$

The measured values can thereby be given by the vector

$$\boldsymbol{b} = \hat{\mathcal{A}}\left(\boldsymbol{M}\right) \in \mathbb{R}^N \ .$$

The special case, in which a subset

$$\Omega \subseteq \mathbb{N}_m \times \mathbb{N}_n = \{(i,j) \mid i \in \mathbb{N}_m, j \in \mathbb{N}_n\} \qquad\qquad |\Omega| = N$$

of entries of $\boldsymbol{M}$ are known, is referred to as the *matrix completion* setting. A corresponding measurement operator $\hat{\mathcal{A}}_\Omega : \mathbb{R}^{m\times n} \to \mathbb{R}^N$ is defined using $\mathcal{A}_{k,i,j} := \delta_{\omega_k,(i,j)}$, i.e.

$$\hat{\mathcal{A}}_\Omega(\boldsymbol{X})[k] := \sum_{i=1}^{m}\sum_{j=1}^{n} \delta_{\omega_k,(i,j)}\boldsymbol{X}[i,j] \ , \tag{5.1}$$

where δ is the Kronecker-delta and $\omega_k \in \Omega$ are the measured positions in an arbitrary but fixed order. In this setting, it is also convenient to define the projection operator

$$\hat{P}_\Omega\left(\boldsymbol{X}\right)[i,j] = \hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\boldsymbol{X})\Big)[i,j] = \begin{cases} \boldsymbol{X}[i,j] & (i,j) \in \Omega \\ 0 & \text{else} \end{cases}$$

onto the space of matrices that are zero except for the measured positions.

In the following, it is assumed that the measurements are incomplete in the sense that $N < mn$, e.g. for matrix completion not all entries are known. The goal is to reconstruct the unknown matrix $\boldsymbol{M}$ from the measurement operator $\hat{\mathcal{A}}$ and the corresponding measurements $\boldsymbol{b} = \hat{\mathcal{A}}(\boldsymbol{M})$. In other words,

$$\hat{\mathcal{A}}\left(\boldsymbol{X}\right) = \boldsymbol{b} \tag{5.2}$$

has to be solved for $\boldsymbol{X} \in \mathbb{R}^{m\times n}$. Without further restrictions, this problem is of course ill-posed, because the space of $m \times n$ matrices is mn dimensional and (5.2) is a system of only $N < mn$ linear equations, yielding infinitely many solutions. In particular, one can define the set of admissible solutions as

$$\operatorname{Adm}\left(\hat{\mathcal{A}}, \boldsymbol{b}\right) := \left\{\boldsymbol{X} \in \mathbb{R}^{m\times n} \;\middle|\; \hat{\mathcal{A}}(\boldsymbol{X}) = \boldsymbol{b}\right\} .$$

Using that $\hat{\mathcal{A}}$ is linear, it is easy to show that $\operatorname{Adm}(\hat{\mathcal{A}}, \boldsymbol{b})$ is a convex set, i.e. for every $0 \le \alpha \le 1$ and $\boldsymbol{X}_1, \boldsymbol{X}_2 \in \operatorname{Adm}(\hat{\mathcal{A}}, \boldsymbol{b})$,

$$\hat{\mathcal{A}}(\alpha\boldsymbol{X}_1 + (1-\alpha)\boldsymbol{X}_2) = \alpha\hat{\mathcal{A}}(\boldsymbol{X}_1) + (1-\alpha)\hat{\mathcal{A}}(\boldsymbol{X}_2) = \alpha\boldsymbol{b} + (1-\alpha)\boldsymbol{b} = \boldsymbol{b}$$

holds and therefore, $\alpha\boldsymbol{X}_1 + (1-\alpha)\boldsymbol{X}_2 \in \operatorname{Adm}(\hat{\mathcal{A}}, \boldsymbol{b})$. It is clear that one of the matrices in $\operatorname{Adm}(\hat{\mathcal{A}}, \boldsymbol{b})$ is equal to the unknown matrix $\boldsymbol{M}$, but without further information it is impossible to infer which. The key assumption, allowing the reconstruction problem to be well-posed, is that the unknown matrix $\boldsymbol{M}$ has a small rank r. From Proposition 3.9, it is known that a matrix of rank r has only $r(m+n-r)$ degrees of freedom. Therefore, if the unknown matrix $\boldsymbol{M}$ has rank r, then $r(m+n-r)$ measurements can in principle be sufficient to determine $\boldsymbol{M}$ as the unique matrix with rank at most r that is compatible with the measured values. In other words, we want that

$$\hat{\mathcal{A}}\left(\boldsymbol{X}\right) \cap \mathcal{M}_{\le r}(\mathbb{R}^{m\times n}) = \{\boldsymbol{M}\} \ . \tag{5.3}$$

As an example, consider the following measured entries of an unknown rank two matrix $\boldsymbol{M}$

$$\begin{pmatrix} 1 & 1 & ? & 1 & 1 \\ 1 & 2 & 1 & ? & ? \\ 1 & 3 & ? & 1 & ? \\ 1 & 4 & ? & ? & 4 \\ 1 & 5 & 1 & ? & ? \end{pmatrix} \in \mathbb{R}^{5\times 5} \ .$$

Here $r(m+n-r) = 16$ entries are known and indeed there is only one possible rank two completion, namely

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 2 \\ 1 & 3 & 1 & 1 & 3 \\ 1 & 4 & 1 & 1 & 4 \\ 1 & 5 & 1 & 1 & 5 \end{pmatrix} \in \mathbb{R}^{5\times 5} \ . \tag{5.4}$$

This is due to the fact, that the last three columns have to be linear combinations of the first two, in order to give a rank two matrix. The resulting sets of linear equations allow the rows of (5.4) as the only solution. Note, however, that having at least $r(m + n - r)$ measurements is no sufficient criterion for uniqueness, as is easily seen in the above example, e.g. by removing one measured entry from the fifth row and adding it anywhere in the third one.

In most practical applications, the actual rank of $\boldsymbol{M}$ is unknown and there is only the somewhat vague assumption that the rank is low. To circumvent this lack of knowledge, one searches the set of admissible matrices for the matrix $\boldsymbol{X}_* \in \operatorname{Adm}(\hat{\mathcal{A}}, \boldsymbol{b})$ with the smallest rank, as formalized in the following.

Problem 5.1 (Matrix Recovery)**.** Let $\hat{\mathcal{A}} : \mathbb{R}^{m \times n} \to \mathbb{R}^N$ be a linear measurement operator and let $\boldsymbol{b} := \hat{\mathcal{A}}(\boldsymbol{M})$ be the corresponding measurements of an otherwise unknown matrix $\boldsymbol{M} \in \mathbb{R}^{m \times n}$. To reconstruct $\boldsymbol{M}$, solve the following optimization problem in $\boldsymbol{X} \in \mathbb{R}^{m \times n}$:

$$\begin{gathered}\text{minimize } \operatorname{rank}(\boldsymbol{X}) \\ \text{subject to } \hat{\mathcal{A}}(\boldsymbol{X}) = \boldsymbol{b}\end{gathered}$$

Note that if (5.3) holds, i.e. if $\boldsymbol{M}$ is the only admissible matrix with rank at most r, then the rank minimization yields $\boldsymbol{M}$ as the unique solution and vice versa. Hence, advance knowledge of the correct rank is not required for this approach. However, this advantage comes at a cost, as the rank minimization has the major drawback that it is NP-hard in general. For the matrix recovery problem, this can be proven by showing that the *vector cardinality minimization* can be recast to a special case of it, see [48]. The vector cardinality minimization problem is the task of finding the sparsest vector compatible with a set of linear measurements and is shown to be NP-hard by Natarajan [115]. For the completion problem, see e.g. [49, 116, 117] and references therein. As a consequence, directly solving the rank minimization is infeasible for large matrices. A popular resort is to instead regard a convex relaxation, namely

Problem 5.2 (Convex Matrix Recovery)**.** In the setting of Problem 5.1, to reconstruct $\boldsymbol{M}$, solve the following convex optimization problem in $\boldsymbol{X} \in \mathbb{R}^{m \times n}$,

$$\begin{gathered}\text{minimize } \|\boldsymbol{X}\|_* \\ \text{subject to } \hat{\mathcal{A}}(\boldsymbol{X}) = \boldsymbol{b}\ .\end{gathered}$$

That is, the minimization of the rank is replaced by the minimization of the nuclear norm. While the rank is the number of non-zero singular values, the nuclear norm is equal to their sum. Therefore, replacing the NP-hard rank minimization by minimization of the nuclear norm is in clear analogy to the classical compressive sensing, where the minimization of the "l_0-norm"[1] is replaced by the convex l_1-norm, see e.g. [118]. The idea to approximate the

[1]Note that in contradiction to its name, the "l_0-norm" is not a norm.

NP-hard rank optimization by the convex nuclear norm optimization is due to Fazel [119] and is a generalization of the trace minimization that was already used to approximate the rank minimization for symmetric, positive semi-definite matrices, see for example the work of Mesbahi and Papavassilopoulos [120]. In some respects, the nuclear norm is the tightest convex relaxation of the rank minimization problem, as it is shown by Fazel et al. [121] that the nuclear norm is the convex envelope[2] of the rank function on the set of matrices with spectral norm bounded by one. Since every norm is a convex function and since the set of admissible matrices is a convex set, Problem 5.2 constitutes a convex optimization problem. As such, it poses a number of practical advantages and can in particular be solved directly by semidefinite programming (see Section 5.1.3). The important question is, under what conditions Problem 5.2 is equivalent to the rank minimization, in the sense that it yields the same solutions. This question was studied by several authors and the most important results are presented in Section 5.1.2.

Finally, the robustness of the recovery is of interest as well. Real-world measurements are usually not perfect and the measured values thus contain a certain error. Additionally, in numerous practical applications, the unknown matrix is only approximately of low rank. In this case, one aims to reconstruct the low rank approximation, as the exact matrix is inaccessible by an incomplete set of measurements. Both scenarios can be modeled by assuming that there is a low rank matrix $\boldsymbol{M}$ that is to be reconstructed, but that the measurements are corrupted by a full rank noise term $\boldsymbol{E} \in \mathbb{R}^{m\times n}$. The measurements are then given as

$$\boldsymbol{b} = \hat{\mathcal{A}}(\boldsymbol{M}) + \hat{\mathcal{A}}(\boldsymbol{E}) = \hat{\mathcal{A}}(\boldsymbol{M} + \boldsymbol{E}) \ .$$

For an analysis of this problem, it is necessary that the amplitude of the measured noise is bounded. A convenient choice is for example that $\|\hat{\mathcal{A}}^T(\hat{\mathcal{A}}(\boldsymbol{E}))\|_S < \epsilon$ holds for an $\epsilon > 0$, where $\|\cdot\|_S$ is the spectral norm. In order to account for this error, the restrictions on the set of admissible matrices have to be relaxed. This is done by replacing the equality constraint by a quadratic inequality constraint, yielding the error aware formulations of the convex reconstruction problems.

Problem 5.3 (Convex Matrix Recovery with Error)**.** Let $\hat{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^N$ be a linear measurement operator and let $\boldsymbol{b} := \hat{\mathcal{A}}(\boldsymbol{M} + \boldsymbol{E})$ be the corresponding noisy measurements of an otherwise unknown matrix $\boldsymbol{M} \in \mathbb{R}^{m\times n}$. Assume that the noise term is bounded by $\|\hat{\mathcal{A}}^T(\hat{\mathcal{A}}(\boldsymbol{E}))\|_S < \epsilon$ for an $\epsilon > 0$. To reconstruct $\boldsymbol{M}$, solve the following optimization problem in $\boldsymbol{X} \in \mathbb{R}^{m\times n}$:

$$\begin{aligned} \text{minimize } & \|\boldsymbol{X}\|_* \\ \text{subject to } & \left\|\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}\left(\boldsymbol{X}\right) - \boldsymbol{b}\right)\right\| < \epsilon \ . \end{aligned}$$

The choice of the norm $\|\cdot\|$ depends on the setting. In the general recovery setting the spectral norm is convenient, while in the completion setting the Frobenius norm can be used.

[2]The convex envelope of a function $f : D \to \mathbb{R}$ is defined as the largest convex function $g : D \to \mathbb{R}$, such that $g(x) \leq f(x)$ holds for all $x \in D$.

5.1.2 Theoretical Results for Matrix Recovery and Completion

The reconstruction of matrices from incomplete data has already been studied for several decades. However, in the early studies this was usually not done in the mindset of recovering a matrix from incomplete measurements. Instead, the goal often was to approximate matrices with a specific structure, or to understand under which conditions a compatible matrix with certain properties exists. The settings considered are not limited to finding compatible matrices with minimal rank, but also include settings in which compatible matrices with different properties are desired. A worthwhile survey of these very general matrix reconstruction settings was done by Johnson [122]. Early works in the setting of low rank reconstruction include for example one by Gabriel and Zamir [123] from 1979, who studied the low-rank approximation of matrices with respect to a variation of the Frobenius norm, where each entry is considered according to a weight assigned to it. This contains the matrix completion setting as a special case with weights equal to zero or one. Another example is the work of Woerdeman [124], who studied the low rank reconstruction of block matrices, where only the lower triangular part of the matrix is known. The major shift to consider low rank techniques as a means to recover matrices from incomplete measurements is due to the wide success of compressive sensing. Carrying over ideas and concepts from compressive sensing also led to substantive progress in matrix recovery.

One essential tool adapted from compressive sensing is the *restricted isometry property* (RIP), which allows the formulation of sufficient criteria for unique reconstruction. The RIP was first introduced in classical compressive sensing by Candès and Tao [125] and then transferred to the matrix recovery problem by Recht et al. [48]. In its formulation for matrix recovery, it is defined as follows.

Definition 5.4 (Restricted Isometry Property [48])**.** Let $\hat{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^N$ be a linear operator. For every natural number $1 \le r \le \min(m, n)$, the r-restricted isometry constant (r-RIC) of $\hat{\mathcal{A}}$, is defined as the smallest number $\delta_r(\hat{\mathcal{A}})$, such that

$$\left(1 - \delta_r(\hat{\mathcal{A}})\right)\|\boldsymbol{X}\|_F \le \left\|\hat{\mathcal{A}}(\boldsymbol{X})\right\|_2 \le \left(1 + \delta_r(\hat{\mathcal{A}})\right)\|\boldsymbol{X}\|_F \ , \tag{5.5}$$

holds for all matrices $\boldsymbol{X}$ of rank at most r. An operator $\hat{\mathcal{A}}$ is said to obey the RIP of order r with RIC $\delta_r(\hat{\mathcal{A}})$, if $\delta_r(\hat{\mathcal{A}})$ is smaller than one.

Note that by construction, for every measurement operator $\hat{\mathcal{A}}$, it holds that $\delta_{r'} \le \delta_r$ for $r' < r$. As a first result using the RIP, Recht et al. [48] established the following theorem.

Theorem 5.5 (Recht et al. [48, Theorem 3.3])**.** *Let $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ be a matrix of rank r and let $\hat{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^N$ be a measurement operator with restricted isometry constant $\delta_{5r} < 1/10$. Then, $\boldsymbol{M}$ can be recovered exactly from the corresponding measurements $\boldsymbol{b} = \hat{\mathcal{A}}(\boldsymbol{M})$, as it is the unique minimum of the convex optimization Problem 5.2.*

This result is substantially improved by Candès and Plan [126], who proved the following theorem, which additionally ensures that the recovery is stable in the presence of noise.

Theorem 5.6 (Candès and Plan [126, Theorem 2.4])**.** *Let $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ be a matrix of rank r, let $\hat{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^N$ be a measurement operator and let*

$$\boldsymbol{b} = \hat{\mathcal{A}}(\boldsymbol{M} + \boldsymbol{E})$$

be a corresponding noisy measurement of $\boldsymbol{M}$. For this, let $\boldsymbol{E} \in \mathbb{R}^{m\times n}$ be a noise term with $\|\hat{\mathcal{A}}^T(\hat{\mathcal{A}}(\boldsymbol{E}))\|_S \leq \epsilon$. Assume that $\hat{\mathcal{A}}$ has a restricted isometry constant $\delta_{4r} < \sqrt{2} - 1$. Then, for the minimizer $\boldsymbol{X}_*$ of the convex Problem 5.3,

$$\|\boldsymbol{X}_* - \boldsymbol{M}\|_F \leq C\sqrt{r}\epsilon,$$

holds for a constant C, depending only on the isometry constant $\delta_{4r}(\hat{\mathcal{A}})$.

This theorem shows the power of the restricted isometry property. However in order for this to be useful in practice, it is necessary that there are in fact measurement operators obeying the RIP. This is for example shown by Recht et al. [48], who prove for different sampling methods that random measurement operators obey the RIP with high probability, provided that they correspond to a sufficiently large set of measurements. As an explicit example, their result for Gaussian measurement operators is presented here. A measurement operator

$$\hat{\mathcal{A}}(\boldsymbol{X}) = \sum_{k=1}^{N}\sum_{i=1}^{m}\sum_{j=1}^{n} \mathcal{A}_{k,i,j}\ \boldsymbol{X}[i,j]\ \boldsymbol{e}_k\ ,$$

is completely defined by the coefficients $\mathcal{A}_{k,i,j}$. For given dimensions m, n and a number of measurements N, a Gaussian measurement operator is created by sampling each coefficient independently from identical normal distributions. The result of Recht et al. [48] for Gaussian measurement operators can be stated as follows.

Theorem 5.7 (Recht et al. [48])**.** *Fix $0 < \delta < 1$. If $\hat{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^N$ is sampled as a Gaussian measurement operator, then for every $1 \leq r \leq \min(m, n)$, there exist constants $c_0, c_1 > 0$ depending only on δ such that, with probability at least $1 - e^{-c_1 p}$, the r-RIC of $\hat{\mathcal{A}}$ obeys $\delta_r(\hat{\mathcal{A}}) \leq \delta$, whenever $N \geq c_0 r(m + n)\log(mn)$.*

Altogether, the above results show that exact and robust matrix recovery is indeed possible for a large class of (random) measurement operators, which obey the RIP. Unfortunately, however, the measurement operators corresponding to the matrix completion setting invariably do not obey the RIP of any order. To show this, let Ω be any incomplete set of measured positions and let $\hat{\mathcal{A}}_\Omega$ be the corresponding measurement operator defined as in (5.1). Let $(i, j) \notin \Omega$ be an entry that is not measured and consider the matrix $\boldsymbol{X} = \gamma\boldsymbol{e}_i\boldsymbol{e}_j^T$, which has only this very entry equal to $\gamma > 0$ and all others equal to zero. This is a rank one matrix with $\|\boldsymbol{X}\|_F = \gamma$ and from (5.1), it is clear that $\hat{\mathcal{A}}(\boldsymbol{X}) = \boldsymbol{0}$ holds. From the left part of (5.5), it then follows with

$$(1 - \delta_1(\hat{\mathcal{A}}))\|\boldsymbol{X}\|_F \leq \|\hat{\mathcal{A}}(\boldsymbol{X})\|_2 = 0\ ,$$

that the 1-RIC obeys $\delta_1(\hat{\mathcal{A}}) \geq 1$. Since the restricted isometry constants for larger ranks can only be larger, this shows that $\hat{\mathcal{A}}$ does not obey any RIP. Furthermore, it is easy to see that in the completion setting, one cannot hope to reconstruct *all* matrices from *any* incomplete measurement set. In the example above, all measured entries of $\boldsymbol{X} = \gamma \boldsymbol{e}_i \boldsymbol{e}_j^T$ are zero and therefore, there is absolutely no way to infer the factor γ from the measurements. As a consequence, other criteria for unique reconstruction are needed in the completion setting.

A first rigorous analysis of the matrix completion problem in the formulation used in this chapter was done by Candès and Recht [49]. As no measurement set works for all matrices, they used probabilistic arguments instead. More precisely, they introduced a *random orthogonal model*, which defines a procedure to randomly sample matrices from the set of all matrices with a fixed rank. For given dimensions m, n with $m \leq n$ and rank r, matrices $\boldsymbol{U} \in \mathbb{R}^{m\times r}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times r}$ are independently and uniformly sampled from the sets of all orthogonal $m \times r$ and $n \times r$ matrices, respectively. With an arbitrarily sampled diagonal matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{r\times r}$, a rank r matrix

$$\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \in \mathcal{M}_r\left(\mathbb{R}^{m\times n}\right)$$

is obtained. Their main results is that if the unknown matrix $\boldsymbol{M}$ is sampled from the random orthogonal model and at least

$$|\Omega| = Cn^{1.2} r \log(n)$$

positions are sampled at random by incrementally sampling positions uniformly from all remaining positions, then the minimizers of the convex Problem 5.2 and the rank minimization Problem 5.1 are both unique and equal to $\boldsymbol{M}$, with a probability of at least $1 - cn^{-3}$. Roughly speaking, this result states that most matrices can be recovered from most measurement sets containing $\Theta(n^{1.2} r \log(n))$ entries. This result is already quite powerful, as for low rank r this requires only a tiny fraction of the number of entries, while maintaining an exact reconstruction.

This result was subsequently improved by Candès and Tao [127], Keshavan et al. [116] and Recht [128]. An important property needed to formulate Recht's result is the coherence defined as follows.

Definition 5.8 (Coherence)**.** Let $V \subset \mathbb{R}^n$ be a r-dimensional subspace of $\mathbb{R}^n$ and let $\hat{P}_V$ be the orthogonal projection onto V. The coherence of V with respect to the canonical basis $\{\boldsymbol{e}_i\}_{i=1,\dots,n}$ is defined as

$$\mu(V) := \frac{n}{r} \max_{1\leq i\leq n} \left\|\hat{P}_V(\boldsymbol{e}_i)\right\|_2^2 .$$

The largest possible coherence is n/r, which is achieved if any canonical basis vector is contained in the subspace. The smallest possible coherence of any subspace is 1, which corresponds to a subspace that is spanned by basis vectors whose entries all have the same magnitude $1/\sqrt{n}$. It turns out that in order to reconstruct a matrix from an incomplete set of entries, a low coherence of its row and column spaces is important. Recht [128] argues that if both spaces have a low coherence, then all entries carry approximately the same information and vice versa. To give an intuition to this, consider the matrix used as a counter example before,

$$\boldsymbol{e}_1\boldsymbol{e}_1^T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{4\times 4} .$$

This matrix has maximal coherence, since the first row and column vectors are both equal to $\boldsymbol{e}_1$. Here, it is intuitively clear that the information gained by measuring the top left entry is much higher than measuring any of the other entries. The main result of Recht [128] is the following theorem.

Theorem 5.9 (Recht [128])**.** *Let $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ be a matrix of rank r, with singular value decomposition $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$. Without loss of generality, impose that $m \leq n$. Assume that*

- *the row and column spaces of $\boldsymbol{M}$ have coherence bounded above by some positive μ_0,*
- *the matrix $\boldsymbol{U}\boldsymbol{V}^T$ has a maximum entry bounded by $\mu_1\sqrt{r/mn}$ in absolute value for some positive μ_1.*

Suppose N entries of $\boldsymbol{M}$ are measured with locations Ω sampled uniformly at random. If

$$N \geq 32 \max(\mu_0, \mu_1^2) r(m+n)\beta(\log(2n))^2$$

for a $\beta > 1$, then the minimizer of the convex Problem 5.2 is unique and equal to $\boldsymbol{M}$ with probability at least $1 - 6\log(n)(m+n)^{2-2\beta} - n^{2-2\beta^{1/2}}$.

Asymptotically, the bound is nearly optimal, exceeding the definite lower bound of $r(m+n-r)$ required entries only by a logarithmic factor. It is shown by Candès and Recht [49] that for random matrices, the requirements of Theorem 5.9 are fulfilled with high probability. In particular, they show that with high probability, matrices sampled from the random orthogonal model have a low coherence and that the second assumption is also fulfilled with high probability, with a μ_1 scaling at most logarithmically in n.

Finally, Candès and Plan [129] showed that matrix completion is also robust to noise. Roughly speaking, they show that if the unknown matrix $\boldsymbol{M}$ and the measured locations Ω are such that $\boldsymbol{M}$ can be recovered from noiseless measurements by solving the convex Problem 5.2, then $\boldsymbol{M}$ can also be recovered from noisy measurements by solving Problem 5.3 (using the Frobenius norm), with an error

$$\|\boldsymbol{M} - \boldsymbol{X}_*\|_F \leq C\epsilon\sqrt{\frac{2mn\min(m,n)}{N}}$$

proportional to the noise level $\epsilon = \|P_\Omega(\boldsymbol{E})\|_F$. Here, $\boldsymbol{X}_*$ is the minimizer of Problem 5.3 and C is a constant.

In conclusion of this section, let us note that this altogether shows, that many low rank matrices can indeed be recovered exactly from incomplete subsets of their entries and that this reconstruction is robust with respect to noise. Additionally, the recovery can be performed by solving the convex optimization Problem 5.2, instead of the NP-hard rank minimization Problem 5.1.

5.1.3 Reconstruction Algorithms

Up to this point, only theoretical results are given, guaranteeing the ability to reconstruct matrices from certain incomplete measurements. Therefore, this section gives some examples for the practical realizations of such reconstructions, i.e. algorithms which (approximately) solve the optimization problems introduced in the previous sections. This following collection is not exhaustive and focuses on those algorithms that are also of interest for the more general tensor recovery setting. In their first paper on matrix recovery, Recht et al. [48] show that the convex problem, Problem 5.2, can be solved by semidefinite programming. For the special case of matrix completion, this is already shown by Candès and Recht [49]. The formulation as a semidefinite program has the advantage that well established algorithms like interior point methods are directly available. However, these general algorithms do not take the special properties of these matrix problems into account and it is evident that they have difficulties with large matrices. For example, Cai et al. [117] state that SDPT3 [130], a state of the art semidefinite programming solver, is only capable of handling $n \times n$ matrices with $n \leq 100$. Accordingly, specialized algorithms are required for efficient matrix recovery.

One of the simplest, yet quite powerful, algorithms for matrix reconstruction is the *iterative hard thresholding* (IHT) algorithm, see [131, 132]. The IHT algorithm can be seen as an Riemannian gradient descent on the low rank matrix manifold. In its most basic form, the iteration is given by

$$\boldsymbol{X}_{i+1} := \hat{H}_r\Big(\boldsymbol{X}_i - \alpha\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\boldsymbol{X}_i) - \boldsymbol{b}\Big)\Big) \ ,$$

where $\hat{H}_r$ is the hard-thresholding operator defined in (3.2) and α is a step size. The algorithm is not by itself rank adaptive and requires an a priori guess on the rank. However, rank adaptation strategies are available and presented in the aforementioned papers.

A second iterative approach is the *singular value thresholding* (SVT) algorithm, which is an adaptation of linearized Bregman iterations for matrix reconstruction. Bregman iterations were first introduced by Osher et al. [133] and have successfully been applied to l_1-norm minimization by Yin et al. [134] and others. The adaptation to the matrix reconstruction setting is due to Cai et al. [117]. The SVT algorithm combines a nuclear norm minimization step using singular value soft thresholding with a modified gradient step. Cai et al. [117] show that the resulting iteration converges to the unique minimizer of

$$\begin{aligned} \text{minimize} \quad & \tau \|\boldsymbol{X}\|_* + \frac{1}{2}\|\boldsymbol{X}\|_F^2 \\ \text{subject to} \quad & \hat{\mathcal{A}}(\boldsymbol{X}) = \boldsymbol{b} \ , \end{aligned}$$

where $\tau > 0$ is a parameter of the algorithm and can be chosen freely. Furthermore, they show that for $\tau \to \infty$, this minimizer indeed converges to the minimum norm solution of the convex matrix recovery, i.e. Problem 5.2. The major advantage of the SVT algorithm compared to the IHT algorithm is that it is inherently rank adaptive, i.e. no *a priori* knowledge on the rank is needed. Computationally, the SVT algorithm is very efficient, with Cai et al. [117] being able to reconstruct matrices in the completion setting with rank 10 and dimensions of up to $30,000 \times 30,000$ on an ordinary desktop computer.

Another interesting approach is alternating subspace optimization, which is examined for the matrix recovery setting by Jain et al. [135]. The idea is to impose a low rank decomposition, e.g. $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T, \boldsymbol{U} \in \mathbb{R}^{n\times r}, \boldsymbol{V} \in \mathbb{R}^{m\times r}$, and optimize by alternatingly finding the optimal $\boldsymbol{U}$ while keeping $\boldsymbol{V}$ fixed, and then finding the optimal $\boldsymbol{V}$ while keeping $\boldsymbol{U}$ fixed. This approach can be seen as the order two special case of the more general alternating least squares (ALS) algorithm briefly introduced in Section 4.4 and examined in details by Holtz et al. [97] and Espig et al. [98]. Just like the IHT, this algorithm is not by itself rank adaptive and needs an a priori guess on the rank and possibly additional rank adaptation strategies.

5.2 The Tensor Recovery Setting

This section is concerned with higher order generalizations of the low rank matrix recovery setting from the previous section. In principle, such generalizations are possible using any of the tensor rank definitions introduced in Chapter 3. The canonical format, however, has several undesirable properties making it an unsuitable candidate, especially concerning numerical computations. In particular, the NP-hardness of even determining the rank of a given tensor, as well as the fact that the set of low CP-rank tensors is not closed, pose problems, see Section 3.2 for the details. In the following, we therefore focus on the subspace based tensor rank definitions, in particular the Tensor Train format. The Tensor Train format is used as the prime example, as it combines the favorable features of a generalized SVD and manifold structure with a linear scaling with respect to the order. However, the definitions and results of this section can be straightforwardly adapted to the simpler Tucker format and to large extent also to the HT-format, as they share the subspace based definition.

5.2.1 Formal Definition and Types of Measurements

Let $\mathcal{T} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ be an unknown tensor that shall be reconstructed and assume that N linear measurements

$$\boldsymbol{b}[k] = \sum_{i_1,\ldots,i_d} \mathcal{A}_{k,i_1,\ldots,i_d}\ \mathcal{T}[i_1,\ldots,i_d]$$

are available, for some $\mathcal{A}_{k,i_1,\ldots,i_d} \in \mathbb{R}$. That is, each sample measures a linear superposition of the entries. The corresponding measurement operator $\hat{\mathcal{A}} : \mathbb{R}^{n_1 \times \ldots \times n_d} \to \mathbb{R}^N$ is defined entrywise as

$$\hat{\mathcal{A}}(\mathcal{X})[k] := \sum_{i_1,\ldots,i_d} \mathcal{A}_{k,i_1,\ldots,i_d}\ \mathcal{X}[i_1,\ldots,i_d] \tag{5.6}$$

and its adjoint operator $\hat{\mathcal{A}}^T : \mathbb{R}^N \to \mathbb{R}^{n_1 \times \ldots \times n_d}$ as

$$\hat{\mathcal{A}}^T(\boldsymbol{v})[i_1,\ldots,i_d] := \sum_{k=1}^{N} \mathcal{A}_{k,i_1,\ldots,i_d}\ \boldsymbol{v}[k]\ .$$

Note that, even though for a fixed rank the TT-format scales only linearly in the order, one cannot hope to solve this very general recovery problem in sub-exponential complexity, as the measurement coefficients $\mathcal{A}_{k,i_1,\ldots,i_d}$ themselves can already be interpreted as a tensor $\mathcal{A} \in \mathbb{R}^{N \times n_1 \times \ldots \times n_d}$, amounting to a storage complexity of $\mathcal{O}(n^d N)$. Hence, we are mostly interested in special cases, for which we can hope to avoid the exponential complexity. In this work, we propose the concept of *rank-one samples*, which represent such a special case not suffering from the curse of dimensionality. Rank-one samples are samples, for which the measurement coefficients can be expressed as

$$\mathcal{A}_{k,i_1,\ldots,i_d} = \boldsymbol{a}_{1,k}[i_1]\ \boldsymbol{a}_{2,k}[i_2] \ldots \boldsymbol{a}_{d,k}[i_d]\ ,$$

for some $\boldsymbol{a}_{\mu,k} \in \mathbb{R}^{n_\mu}$. Interpreting $\mathcal{A} \in \mathbb{R}^{N \times n_1 \times \ldots \times n_d}$ as a tensor, this means that there exists a rank N CP-decomposition such that

$$\mathcal{A} = \sum_{k=1}^{N} \boldsymbol{e}_k \otimes \boldsymbol{a}_{1,k} \otimes \boldsymbol{a}_{2,k} \otimes \ldots \otimes \boldsymbol{a}_{d,k}\ . \tag{5.7}$$

In this way, $\mathcal{A}$ itself has a storage complexity of $\mathcal{O}(dnN)$. The concept of rank-one samples is more general that the popular completion setting, in which only certain entries of the tensor are known. That is, a set of N entries of $\mathcal{T}$ with positions

$$\Omega \subseteq \bigtimes_{k=1}^{d} \mathbb{N}_{n_k} = \{(i_1,\ldots,i_d) \mid i_\mu \in \mathbb{N}_{n_\mu},\ \mu = 1,\ldots,d\}$$

is measured. To see that the completion setting is indeed a special case of rank-one samples, denote the positions as $\{(j_{1,1},\ldots,j_{d,1}),\ldots,(j_{1,N},\ldots,j_{d,N})\}$. Then, a corresponding rank one measurements is given by

$$\mathcal{A} = \sum_{k=1}^{N} \boldsymbol{e}_k \otimes \boldsymbol{e}_{j_{1,k}} \otimes \boldsymbol{e}_{j_{2,k}} \otimes \ldots \otimes \boldsymbol{e}_{j_{d,k}}\ ,$$

where the $\boldsymbol{e}_{j_{\mu,k}} \in \mathbb{R}^{n_\mu}$ are the canonical basis vectors. In the tensor completion setting, it is sometimes also convenient to use the projection operator

$$\hat{P}_\Omega\left(\mathcal{X}\right)[i_1,\dots,i_d] = \hat{\mathcal{A}}_\Omega^T\left(\hat{\mathcal{A}}_\Omega\left(\mathcal{X}\right)\right)[i_1,\dots,i_d] = \begin{cases} \mathcal{X}[i_1,\dots,i_d] & \text{if } (i_1,\dots,i_d)\in\Omega \\ 0 & \text{else} \end{cases}$$

onto the set of tensors that are zero everywhere except at the measured positions and the corresponding tensor

$$\mathcal{B} := \hat{P}_\Omega\left(\mathcal{T}\right)$$

containing only the measured entries.

Analogously to the matrix recovery problem, the goal is to reconstruct $\mathcal{T} \in \mathbb{R}^{n_1\times\ldots\times n_d}$ from the measurements $\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T}) \in \mathbb{R}^N$. For given measurements, the set of admissible tensors can be defined as

$$\operatorname{Adm}\left(\hat{\mathcal{A}},\boldsymbol{b}\right) := \left\{\mathcal{X}\in\mathbb{R}^{m\times n} \;\middle|\; \hat{\mathcal{A}}(\mathcal{X}) = \boldsymbol{b}\right\} .$$

As with matrices, $\operatorname{Adm}\left(\hat{\mathcal{A}},\boldsymbol{b}\right)$ is convex, because for any $\beta\in[0,1]$ and any $\mathcal{X}_1,\mathcal{X}_2 \in \operatorname{Adm}\left(\hat{\mathcal{A}},\boldsymbol{b}\right)$ it holds that

$$\hat{\mathcal{A}}(\beta\mathcal{X}_1 + (1-\beta)\mathcal{X}_2) = \beta\hat{\mathcal{A}}(\mathcal{X}_1) + (1-\beta)\hat{\mathcal{A}}(\mathcal{X}_2) = \beta\boldsymbol{b} + (1-\beta)\boldsymbol{b} = \boldsymbol{b}$$

and therefore, $\beta\mathcal{X}_1 + (1-\beta)\mathcal{X}_2 \in \operatorname{Adm}(\hat{\mathcal{A}},\boldsymbol{b})$. Obviously, the sought tensor $\mathcal{T}$ is contained in $\operatorname{Adm}\left(\hat{\mathcal{A}},\boldsymbol{b}\right)$, however, assuming incomplete measurements, i.e. $N < \prod_{\mu=1}^d n_\mu$, there are also infinitely many other admissible tensors, as $\hat{\mathcal{A}}(\mathcal{X}) = \boldsymbol{b}$ is a system of N linear equations in a $\prod_{\mu=1}^d n_\mu$-dimensional space. Without further restrictions, the recovery problem is therefore fundamentally ill-posed. As mentioned in the beginning, we use the assumption that $\mathcal{T}$ has a small TT-rank $\boldsymbol{r} = (r_1,\dots,r_{d-1})$. Similar to the matrix case, this could in principle allow a reconstruction of $\mathcal{T}$ from highly incomplete measurements, as the TT-manifold of rank $\boldsymbol{r}$ has a dimension $\sum_{\mu=1}^d r_{\mu-1}n_\mu r_\mu - \sum_{\mu=1}^{d-1} r_\mu^2$, where as a convention $r_0 = r_d = 1$. As a consequence, $\mathcal{O}(dnr^2)$ measurements, where $r = \max(r_\mu)$ and $n = \max(n_\mu)$, could be sufficient to determine $\mathcal{T}$ as the single admissible tensor with TT-rank at most $\boldsymbol{r}$, i.e.

$$\operatorname{Adm}\left(\hat{\mathcal{A}},\boldsymbol{b}\right) \cap \mathcal{M}_{\preceq\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\ldots\times n_d}) = \{\mathcal{T}\} . \tag{5.8}$$

5.2.2 Reconstruction Approaches

There are several different ways to approach the tensor recovery problem. The most straightforward approach is to simply use one of the matricizations of $\mathcal{T}$. If $\mathcal{T}$ has TT-rank $(r_1,\dots,r_{d-1})$, then Corollary 3.21 provides that the matricizations $\boldsymbol{T}^{<\mu>}$ have rank r_μ, i.e. a low TT-rank directly implies a low rank of these matricizations. The measurement operator $\hat{\mathcal{A}}$ can straightforwardly be transformed to an operator acting on the matricizations and thus, the complete machinery of matrix recovery introduced in

Section 5.1 can be applied to reconstruct $\mathcal{T}$ through one of its matricizations. In particular, recovery guarantees from the matrix theory directly extend to the tensor setting in this way. However, a simple parameter counting argument suggest that this is not the best approach. Due to its rank, the μ-th matricization has

$$\dim\left(\mathcal{M}_{r_\mu}(\mathbb{R}^{n_1\cdot\ldots\cdot n_\mu \times n_{\mu+1}\cdot\ldots\cdot n_d})\right) = r_\mu\left(\prod_{\nu=1}^{\mu} n_\nu + \prod_{\nu=\mu+1}^{d} n_\nu - r_\mu\right) \tag{5.9}$$

degrees of freedom. This gives a definite lower bound for the number of measurements needed. In almost all cases, this is much more than the

$$\dim\left(\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\ldots\times n_d})\right) = \sum_{\nu=1}^{d} r_{\nu-1}n_\nu r_\nu - \sum_{\nu=1}^{d-1} r_\nu^2$$

degrees of freedom that $\mathcal{T}$ has according to its TT-rank (see Theorem 3.23). In particular, the required number of measurements imposed by (5.9) scale exponentially in the order, because either $\prod_{\nu=1}^{\mu} n_\nu$ or $\prod_{\nu=\mu+1}^{d} n_\nu$ has to scale as $\Theta(n^{d/2})$. In contrast, the degrees of freedom only scale as $\mathcal{O}(dnr^2)$ judging from the TT-rank. Therefore, in order to obtain a linear scaling in the order, one requires an approach which is based on the complete TT-rank, rather than using one of its entries via a single matricization.

As with matrices, it would be favorable to have an inherently rank adaptive formulation for tensor reconstruction, which does not require an *a priori* guess for the TT-rank. In the matrix setting, this was achieved by minimizing the rank on the set of admissible matrices, as formalized in Problem 5.1. Formally, one can define an analogous optimization problem,

$$\begin{aligned} &\text{minimize } \ \mathrm{TT\text{-}rank}(\mathcal{X}) \\ &\text{subject to } \hat{\mathcal{A}}(\mathcal{X}) = \boldsymbol{b}\ , \end{aligned} \tag{5.10}$$

where the minimization is with respect to the partial order $\preceq$ from Definition 3.12. As with matrices, the solution is in general not unique, as there can be several tensors with the same (minimal) rank. Additionally, as $\preceq$ is only a partial order, there is also the possibility that there are several solutions with different ranks. In particular, there could be two (or more) minimal elements, but no least element, i.e. admissible tensors $\mathcal{X}$ and $\bar{\mathcal{X}}$ with TT-ranks $\boldsymbol{r}$ and $\bar{\boldsymbol{r}}$ respectively, such that

$$\left\{\mathcal{Y}\in\mathrm{Adm}\left(\hat{\mathcal{A}},\boldsymbol{b}\right)\,\middle|\,\mathrm{TT\text{-}rank}(\mathcal{Y})\preceq\boldsymbol{r}\right\} = \emptyset = \left\{\mathcal{Y}\in\mathrm{Adm}\left(\hat{\mathcal{A}},\boldsymbol{b}\right)\,\middle|\,\mathrm{TT\text{-}rank}(\mathcal{Y})\preceq\bar{\boldsymbol{r}}\right\}$$

but $r_\mu < \bar{r}_\mu$ and $r_\nu > \bar{r}_\nu$ for some μ, ν. Therefore, even if (5.8) holds, this optimization might fail to have a unique solution. A possible alternative is to interpret the low TT-rank as multiple separate structures, one for each involved matricization, and introduce a function $F : \mathbb{R}^{d-1} \to \mathbb{R}$ to map the corresponding ranks to the real numbers and minimize according to this function, i.e.

$$\begin{aligned} &\text{minimize } \ F(\mathrm{TT\text{-}rank}(\mathcal{X})) \\ &\text{subject to } \hat{\mathcal{A}}(\mathcal{X}) = \boldsymbol{b}\ . \end{aligned} \tag{5.11}$$

Using this optimization problem, $\mathcal{T}$ is reconstructable, i.e. the unique solution, if and only if

$$\operatorname{Adm}\Big(\hat{\mathcal{A}}, \boldsymbol{b}\Big) \cap \{\mathcal{Y} \in \mathbb{R}^{n_1 \times \ldots \times n_d} \mid F(\text{TT-rank}(\mathcal{Y})) \leq F(\text{TT-rank}(\mathcal{T}))\} = \{\mathcal{T}\} \ .$$

A popular choice for F is a (weighted) sum of the ranks, i.e.

$$F(\boldsymbol{r}) = \sum_{\mu=1}^{d-1} \lambda_\mu r_\mu \ ,$$

with positive $\lambda \in \mathbb{R}$. It is noteworthy that, using strictly positive weights, this preserves all unique solutions of the original problem.

Proposition 5.10. *If $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ is the unique solution of Problem* (5.10)*, then it is also the unique solution of Problem* (5.11) *for any weighted sum F with strictly positive weights.*

Proof. Assume towards contradiction, that $\mathcal{Y} \neq \mathcal{X}$ is a solution of Problem (5.11) with TT-rank $\boldsymbol{r}'$. This implies that

$$F(\boldsymbol{r}') = \sum_{\mu=1}^{d-1} \lambda_\mu r'_\mu \leq \sum_{\mu=1}^{d-1} \lambda_\mu r_\mu = F(\boldsymbol{r})$$

and therefore, that either $\boldsymbol{r}' = \boldsymbol{r}$ or $r'_\mu < r_\mu$ for at least one μ holds. In both cases, $\mathcal{Y}$ is also a solution to (5.10), which would therefore be non-unique in contradiction to the assumption that $\mathcal{X}$ is the unique solution. In this first case, this is obvious, as $\mathcal{Y}$ is admissible and has the same rank as the solution $\mathcal{X}$. For the latter case, note that there can be no admissible tensor $\mathcal{Z}$ with rank $\boldsymbol{r}'' \preceq \boldsymbol{r}'$, as this would imply $F(\boldsymbol{r}'') < F(\boldsymbol{r}')$ and thereby that $\mathcal{Z}$ and not $\mathcal{Y}$ is a solution of (5.11). $\square$

Other choices for F are possible as well, for example using the dimension of the corresponding manifold

$$F(\boldsymbol{r}) = \sum_{i=1}^{d} r_{i-1} n_i r_i - \sum_{i=1}^{d-1} r_i^2 \ , \tag{5.12}$$

minimizes the actual degrees of freedom.[3]

The main problem is that the minimization (5.11) is still NP-hard for most choices of F, because the matrix recovery rank minimization is contained as a special case. This is in particular true for any weighted sum and for (5.12), which are both equivalent to Problem 5.1 for $d = 2$. For matrices, the remedy is to replace the NP-hard rank minimization with the convex nuclear norm minimization. As (5.11) is a minimization over the ranks of several matricizations, a tempting approach to avoid the NP-hardness, is to use the nuclear norms of the respective matricization, instead of the ranks. The minimization thus becomes

$$\begin{aligned} &\text{minimize } F\Big(\|\boldsymbol{X}^{<1>}\|_*, \ldots, \|\boldsymbol{X}^{<d-1>}\|_*\Big) \\ &\text{subject to } \hat{\mathcal{A}}(\mathcal{X}) = \boldsymbol{b} \ . \end{aligned} \tag{5.13}$$

[3]Again, we use the convention that $r_0 = r_d = 1$.

It is easily verified that $\|\mathrm{Mat}_{(1,\dots,\mu)}(\cdot)\|_*$ defines a norm on $\mathbb{R}^{n_1\times\dots\times n_d}$. Therefore, if F is a convex function, then (5.13) is a convex optimization problem. This is in particular true for any weighted sum and for (5.12). For order two, both are also equivalent to the convex matrix recovery, Problem 5.2. In the early works on tensor recovery, this approach to obtain a convex tensor recovery optimization was often used for the Tucker format. The setting for the Tucker format is completely analogous, with the only difference that instead of the TT-rank, the Tucker-rank tuple and the respective nuclear norms have to be minimized. A weighted sum of the (Tucker) nuclear norms is for example used by Liu et al. [42], Gandy et al. [55], Tomioka et al. [136], and Kreimer and Sacchi [137] for tensor recovery. A more general discussion of sums of (Tucker) nuclear norms as a generalization of the matrix nuclear norm is done by Signoretto et al. [138]. In the TT-format, weighted sums of nuclear norms are for example used by Phien et al. [139] for tensor recovery. While successful reconstruction from incomplete measurements is indeed observed given a sufficiently large number of measurements, the theoretical as well as practical results for both formats are not completely satisfactory. Especially the number of samples needed for successful reconstruction exceeds the degrees of freedom by far, e.g. for the Tucker format mostly of order $\Omega(rn^{d-1})$ samples are needed, compared to the $\mathcal{O}(dnr + r^d)$ degrees of freedom of a tensor of Tucker-rank $\boldsymbol{r}$ according to Theorem 3.16, see [140].

The underlying problem here is the convex relaxation. For matrices, it was shown that the nuclear norm is the convex envelope of the rank function on the set of matrices with spectral norm bonded by one. Therefore, replacing the rank function by the nuclear norm gives in some respect the tightest convex relaxation of the rank minimization problem for matrices. Unfortunately, this result does not extend to higher order tensors, as shown by Romera-Paredes and Pontil [141] for the Tucker format. In particular, they show that for $d \geq 3$ the sum of the corresponding nuclear norms is *not* the convex envelope of the sum of the Tucker-rank entries. Much more generally, Oymak et al. [142] show that this general method of forming a convex relaxation cannot yield the desired result. In particular, they show that if the sought-after object is structured in several ways, where each structure can be promoted by minimizing an associated norm, then minimizing any combination these norms, one can do no better order-wise than using only one of the present structures, in terms of measurements needed. As a special case, this includes our setting of tensors, which are structured by the small ranks of the different matricizations used in the Tucker or TT format, which can each be promoted by the corresponding nuclear norm. A similar result was also obtained and demonstrated for the tensor recovery in the Tucker format by Mu et al. [140]. A particular consequence of these results is that for the TT-format, using any convex relaxation formed in this way, one requires an exponential number of measurements for successful reconstruction, as was shown at the beginning of this section for the use of a single matricization rank. While these results represent a significant difficulty, it is important to note that they do not imply that no reconstruction efficiently exploiting all sparsity structures is possible. However, such approaches require more involved convex relaxations than the ones obtained by combining the nuclear norms of the matricizations. The question of the convex envelope, i.e. the optimal convex relaxation, for the weighted sum of the ranks is still open, but there are several new convex relaxations being proposed for tensor recovery, for example by Romera-Paredes and Pontil [141] and Yuan and Zhang [143].

In this work, a different approach is chosen to obtain a tractable reconstruction problem. The idea is to split the reconstruction into an optimization on the low rank manifold and a rank adaptation. First, assume that an upper bound $\boldsymbol{r}^* = (r_1^*, \ldots, r_{d-1}^*)$ for the TT-rank $\boldsymbol{r}$ of the unknown tensor $\mathcal{T}$ is known, i.e. $\boldsymbol{r} \preceq \boldsymbol{r}^*$. We say that $\mathcal{T}$ is uniquely reconstructable from this information, if $\mathcal{T}$ is the only admissible tensor with TT-rank $\preceq \boldsymbol{r}^*$, i.e. if

$$\mathrm{Adm}\Big(\hat{\mathcal{A}}, \boldsymbol{b}\Big) \cap \mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}^*}(\mathbb{R}^{n_1 \times \ldots \times n_d}) = \{\mathcal{T}\} \ .$$

A sufficient condition for this can be given using a generalization of the restricted isometry property (RIP) introduced in Definition 5.4 for matrix recovery. A similar extension of the RIP to higher order tensors was first introduced by Rauhut et al. [56] for the Tucker format.

Definition 5.11 (Tensor Restricted Isometry Property (TRIP))**.** Let $\hat{\mathcal{A}} : \mathbb{R}^{n_1 \times \ldots \times n_d} \to \mathbb{R}^N$ be a linear measurement operator. For every rank tuple $\boldsymbol{r} = (r_1, \ldots, r_{d-1})$, the $\boldsymbol{r}$-restricted isometry constant ($\boldsymbol{r}$-RIC) is defined as the smallest number $\delta_{\boldsymbol{r}}(\hat{\mathcal{A}})$, such that

$$(1 - \delta_{\boldsymbol{r}}(\hat{\mathcal{A}}))\|\mathcal{X}\|_F \leq \|\hat{\mathcal{A}}(\mathcal{X})\|_2 \leq (1 + \delta_{\boldsymbol{r}}(\hat{\mathcal{A}}))\|\mathcal{X}\|_F$$

holds for all tensors $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ with TT-rank $\boldsymbol{r}' \preceq \boldsymbol{r}$. $\hat{\mathcal{A}}$ is said to satisfy the tensor restricted isometry property of order $\boldsymbol{r}$ if $\delta_{\boldsymbol{r}}(\hat{\mathcal{A}}) < 1$.

Note that from the definition, it directly follows that $\delta_{\boldsymbol{r}} \leq \delta_{\boldsymbol{r}'}$ for $\boldsymbol{r} \preceq \boldsymbol{r}'$. A sufficient condition for the uniqueness of $\mathcal{T}$ can now be given by the following proposition.

Proposition 5.12. *Let $\hat{\mathcal{A}}$ be a measurement operator that satisfies the TRIP of order $2\boldsymbol{r} = (2r_1, \ldots, 2r_{d-1})$ and let $\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T})$ be the corresponding measurements of a tensor $\mathcal{T}$ with TT-rank $\boldsymbol{r}$. Then, $\mathcal{T}$ is the unique tensor for which*

$$\begin{aligned} \hat{\mathcal{A}}(\mathcal{X}) &= \boldsymbol{b} \\ \textit{and } \ \mathrm{TT\text{-}rank}(\mathcal{X}) &\preceq \boldsymbol{r} \end{aligned} \tag{5.14}$$

holds.

Proof. It is clear that $\mathcal{T}$ is a solution. To show that it is unique, assume that $\mathcal{Y} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ is any solution to (5.14). From Section 3.4.4, we know that the TT-rank of a sum of tensors is bounded by the (entrywise) sum of the ranks. Since TT-rank$(\mathcal{Y}) \preceq \boldsymbol{r}$, it follows that $\mathcal{T} - \mathcal{Y}$ is a tensor of TT-rank at most $2\boldsymbol{r}$. Using the TRIP, it follows

$$0 = \|\boldsymbol{b} - \boldsymbol{b}\|_2 = \|\hat{\mathcal{A}}(\mathcal{T}) - \hat{\mathcal{A}}(\mathcal{Y})\|_2 = \|\hat{\mathcal{A}}(\mathcal{T} - \mathcal{Y})\|_2 \geq \underbrace{(1 - \delta_{2\boldsymbol{r}})}_{>0}\|\mathcal{T} - \mathcal{Y}\|_F \ ,$$

and therefore, $\|\mathcal{T} - \mathcal{Y}\|_F = 0$ and $\mathcal{Y} = \mathcal{T}$ as demanded. □

For the practical applicability of this sufficient condition, it is important that the TRIP is indeed satisfied for practically obtainable measurement operators. For random measurements, this is shown by Rauhut et al. [65], who prove that the TRIP is with high probability satisfied for randomly sampled measurement operators.

Theorem 5.13 (Rauhut et al. [61, 65])**.** *Let $\boldsymbol{r} = (r_1, \ldots, r_d)$ be a rank tuple, let $\epsilon, \delta_{\boldsymbol{r}} \in (0, 1)$ be constants and let $\mathcal{A} \in \mathbb{R}^{N \times n_1 \times \ldots \times n_d}$ be a measurement tensor with entries sampled independently at random from $\mathcal{N}(0, 1/N)$. Then, the measurement operator $\hat{\mathcal{A}} : \mathbb{R}^{n_1 \times \ldots \times n_d} \to \mathbb{R}^N$ induced via* (5.6) *satisfies the TRIP of order $\boldsymbol{r}$ with RIC $\delta_{\boldsymbol{r}}$, with probability at least $1 - \epsilon$, provided that*

$$N \geq C \delta_{\boldsymbol{r}}^{-2} \max \left\{ dnr^2 \log(dr),\ \log(\epsilon^{-1}) \right\} ,$$

where $r = \max_i(r_i)$, $n = \max_i(n_i)$ and C is a universal constant.

Up to the logarithmic factor $\log(dr)$, the number of measurements needed is order-wise already optimal. Together with Proposition 5.12, this theorem shows that low rank tensors can indeed often be reconstructed from a number of random linear measurements, which scales almost linearly in the order.

Unfortunately, these results are for completely dense measurement operators, which by themselves require an exponential complexity in the order. For the tensor completion setting, it can be shown that no TRIP of any order is satisfied.

Proposition 5.14. *Let $\hat{\mathcal{A}}_\Omega$ be the measurement operator induced by a set of measured positions Ω. Then, $\hat{\mathcal{A}}_\Omega$ does not satisfy the TRIP of any order $\boldsymbol{r}$.*

Proof. Let $\mathcal{Y} = \boldsymbol{e}_{i_1} \otimes \ldots \otimes \boldsymbol{e}_{i_d}$ be the TT-rank one tensor, which is non-zero only at a position $(i_1, \ldots, i_d) \notin \Omega$ that is not measured. Then, $\|\mathcal{Y}\|_F = 1$ and $\hat{\mathcal{A}}_\Omega(\mathcal{Y}) = \mathbf{0}$. Consequently, the $(1, \ldots, 1)$-RIC of $\hat{\mathcal{A}}_\Omega$ has to fulfill

$$(1 - \delta_{(1,\ldots,1)}) = (1 - \delta_{(1,\ldots,1)}) \|\mathcal{Y}\|_F \leq \|\hat{\mathcal{A}}_\Omega(\mathcal{Y})\|_2 = 0 ,$$

from which $\delta_{(1,\ldots,1)} \geq 1$ follows. As $(1, \ldots, 1)$ is the minimal TT-rank, for every TT-rank $\boldsymbol{r}$, the $\boldsymbol{r}$-RIC is larger $\delta_{\boldsymbol{r}} \geq \delta_{(1,\ldots,1)} \geq 1$. □

Nevertheless, there are recent results by Ashraphijuo and Wang [67], which show that also in the tensor completion setting, most tensors have a unique low rank completion for most measurement sets of sufficient size. One of the main results is the following theorem.

Theorem 5.15 (Ashraphijuo and Wang [67, Lemma 5.8])**.** *Let $\mathcal{T} \in \mathbb{R}^{n \times n \times \ldots \times n}$ be an order d tensors of TT-rank $(r_1, \ldots, r_{d-1})$ that shall be completed from an incomplete set of know entries. Define $m = \sum_{k=1}^{d-2} r_{k-1} r_k$, $M = n \sum_{k=1}^{d-2} r_{k-1} r_k - \sum_{k=1}^{d-2} r_k^2$ and $r' = \max\left\{\frac{r_1}{r_0}, \frac{r_2}{r_1}, \ldots, \frac{r_{d-2}}{r_{d-3}}\right\}$, where by convention $r_0 = 1$. Assume that $n > \max\{m + d, 400\}$*

and $r' \leq \min(\frac{n}{6}, r_{d-2})$ hold and also let $0 < \epsilon < 1$ be given. Moreover, assume that the sampling probability satisfies

$$p > \frac{1}{n^{d-2}} \max\left\{63 \log\left(\frac{4n}{\epsilon}\right) + 9 \log\left(\frac{8M}{\epsilon}\right) + 18, 6r_{d-2}\right\} + \frac{1}{\sqrt[4]{n^{d-2}}}$$

Then, with probability at least $(1-\epsilon)\Big(1 - \exp(\frac{-\sqrt{n^{d-2}}}{2})\Big)^{n^2}$, $\mathcal{T}$ is uniquely reconstructable.

Compared to Theorem 5.13, the bound on the measurements is much worse, in particular, it is not linear in the order, suggesting room for future improvement.

Now if $\mathcal{T}$ is the unique tensor of TT-rank at most $\boldsymbol{r}^*$ compatible with the measurements, then $\mathcal{T}$ is also the unique solution of the following optimization problem.

$$\begin{aligned} &\text{minimize } \|\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\|_2 \\ &\text{subject to } \text{TT-rank}(\mathcal{X}) \preceq \boldsymbol{r}^* \ . \end{aligned} \tag{5.15}$$

That is, one searches for the tensor with TT-rank bounded by $\boldsymbol{r}^*$, which best fits the measurements. Using the TRIP, it can be shown that this formulation is stable with respect to noisy measurements.

Proposition 5.16. *Let $\mathcal{T}$ be an unknown tensor with TT-rank $\boldsymbol{r} \preceq \boldsymbol{r}^*$ and let $\hat{\mathcal{A}}$ be a measurement operator that fulfills the TRIP of order $2\boldsymbol{r}^*$ with RIC $\delta_{2\boldsymbol{r}^*}$. Assume that the measurements are corrupted by noise, i.e.*

$$\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T} + \mathcal{E}) = \hat{\mathcal{A}}(\mathcal{T}) + \hat{\mathcal{A}}(\mathcal{E}) \ ,$$

for an arbitrary measurement error $\mathcal{E} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ with $\|\hat{\mathcal{A}}(\mathcal{E})\|_2 \leq \epsilon$. Let $\mathcal{T}^*$ be the corresponding solution of Problem (5.15). For the reconstruction error, it holds that

$$\|\mathcal{T}^* - \mathcal{T}\|_F \leq \frac{2\epsilon}{1 - \delta_{2\boldsymbol{r}}} \ ,$$

i.e. the reconstruction error is linearly bounded in the measurement error.

Proof. It holds that

$$\|\hat{\mathcal{A}}(\mathcal{T}) - \boldsymbol{b}\|_2 = \|\hat{\mathcal{A}}(\mathcal{T}) - \hat{\mathcal{A}}(\mathcal{T}) - \hat{\mathcal{A}}(\mathcal{E})\|_2 = \|\hat{\mathcal{A}}(\mathcal{E})\|_2 \leq \epsilon \ .$$

Therefore, the solution $\mathcal{T}^*$ of Problem (5.15) has to fulfill

$$\|\hat{\mathcal{A}}(\mathcal{T}^*) - \boldsymbol{b}\|_2 \leq \epsilon$$

and TT-rank$(\mathcal{T}^*) \preceq \boldsymbol{r}^*$. By assumption, $\mathcal{T}$ has TT-rank bounded by $\boldsymbol{r}^*$ as well. Therefore, using the TRIP and the fact that $\mathcal{T}^* - \mathcal{T}$ has TT-rank at most $2\boldsymbol{r}^*$, it follows that

$$\begin{aligned} \|\mathcal{T}^* - \mathcal{T}\|_F &\leq \frac{1}{1-\delta_{2\boldsymbol{r}}} \|\hat{\mathcal{A}}(\mathcal{T}^* - \mathcal{T})\|_2 \\ &= \frac{1}{1-\delta_{2\boldsymbol{r}}} \|\hat{\mathcal{A}}(\mathcal{T}^*) - \boldsymbol{b} + \boldsymbol{b} - \hat{\mathcal{A}}(\mathcal{T})\| \\ &\leq \frac{1}{1-\delta_{2\boldsymbol{r}}} \Big(\|\hat{\mathcal{A}}(\mathcal{T}^*) - \boldsymbol{b}\| + \|\boldsymbol{b} - \hat{\mathcal{A}}(\mathcal{T})\|\Big) \\ &\leq \frac{2\epsilon}{1-\delta_{2\boldsymbol{r}}} \ , \end{aligned}$$

as claimed. □

The following section introduces methods to solve Problem (5.15) for different types of measurements, as well as strategies to find a rank estimate r^*, should it not be known in advance.

5.3 Algorithmic Treatment of the Tensor Reconstruction Problem

The previous section showed that some of the reconstruction guarantees for matrices can be generalized to higher order tensors. However, in contrast to the matrix recovery setting, finding an optimal convex relaxation of the NP-hard rank minimization problem turns out to be a major difficulty. Circumventing this problem, we assume that an estimate $\boldsymbol{r}$ for the rank is known, thereby transforming the tensor reconstruction problem to an optimization problem on the set $\mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}}(\mathbb{R}^{n_1\times\dots\times n_d})$ of low TT-rank tensors. More precisely, given a measurement operator $\hat{\mathcal{A}} : \mathbb{R}^{n_1\times\dots\times n_d} \to \mathbb{R}^N$ and the corresponding measurements $\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T}) \in \mathbb{R}^N$ of an otherwise unknown tensor $\mathcal{T} \in \mathbb{R}^{n_1\times\dots\times n_d}$, the recovery problem becomes

$$\begin{aligned} \text{minimze } & J(\mathcal{X}) = \left\|\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\right\|_2^2 \\ \text{subject to } & \text{TT-rank}(\mathcal{X}) \preceq \boldsymbol{r} \end{aligned} . \tag{5.16}$$

This section introduces several methods for tackling this optimization problem. The focus is on the alternating steepest descent method presented at the end of the section. For this method, rank adaptation strategies are also introduced, which provide estimates for the rank of the unknown tensor, if none is available a priori.

5.3.1 Iterative Hard Thresholding

Iterative hard thresholding (IHT) is a rather simple, yet quite powerful, method for tensor recovery. The IHT principle is successfully used in classical compressive sensing, as well as matrix completion, see e.g. the work of Tanner and Wei [132] and Blumensath and Davies [144]. For tensor recovery, the IHT algorithm was introduced and refined by Rauhut et al. [56, 61, 65]. Notably, Rauhut et al. [65] were able to prove local recovery guarantees for the IHT in the tensor recovery setting using the TRIP.

The fundamental idea behind the IHT algorithm is to combine a gradient descent iteration with a rank truncation in each step. Using the fixed rank optimization formulation (5.16), the gradient $\mathcal{G}$ is given by

$$\mathcal{G}(\mathcal{X}) := \boldsymbol{\nabla} J(\mathcal{X}) = \boldsymbol{\nabla}\left\|\hat{\mathcal{A}}\left(\mathcal{X}\right) - \boldsymbol{b}\right\|_2^2 = 2\,\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\right) .$$

Starting at some $\mathcal{X}_0$, the iterations of the IHT algorithm are defined recursively as

$$\mathcal{X}_{i+1} := \hat{H}_{\boldsymbol{r}}\left(\mathcal{X}_i - \beta_i \mathcal{G}(\mathcal{X}_i)\right) = \hat{H}_{\boldsymbol{r}}\left(\mathcal{X}_i - \beta_i \hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X}_i) - \boldsymbol{b}\right)\right) ,$$

where $\hat{H}_{\boldsymbol{r}} : \mathbb{R}^{n_1\times\ldots\times n_d} \to \mathcal{M}^{\mathrm{TT}}_{\preceq\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ is a hard thresholding operator and β_i is a sequence of step sizes. Ideally, one wants the hard thresholding operator $\hat{H}_{\boldsymbol{r}} : \mathbb{R}^{n_1\times\ldots\times n_d} \to \mathcal{M}^{\mathrm{TT}}_{\preceq\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ to use the best rank $\boldsymbol{r}$ approximation, i.e.

$$\hat{H}_{\boldsymbol{r}}(\mathcal{X}) = \underset{\mathcal{Y}\in\mathcal{M}^{\mathrm{TT}}_{\preceq\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})}{\operatorname{argmin}} \|\mathcal{X} - \mathcal{Y}\| .$$

However, as finding this best approximation is NP-hard in general (see Section 3.4.2), the quasi-best rank $\boldsymbol{r}$ approximation obtained by a truncated TT-SVD is usually used instead. Obtaining good step sizes for the IHT algorithm is a non-trivial task as well. The locally optimal step sizes are

$$\beta_i^{\mathrm{opt}} = \underset{\beta\in\mathbb{R}}{\operatorname{argmin}}\left\|\hat{\mathcal{A}}\left(\hat{H}_{\boldsymbol{r}}\left(\mathcal{X}_i - \beta\mathcal{G}(\mathcal{X}_i)\right)\right) - \boldsymbol{b}\right\|_2 . \tag{5.17}$$

That is, the step sizes are chosen, such that the residual for the next iterate is minimal. However, to the authors knowledge, there is no analytic solution to (5.17) for any commonly used thresholding operator $\hat{H}_{\boldsymbol{r}}$, which therefore requires rather expensive backtracking or similar approaches. Nevertheless, there are several approximations available, which are obtained by replacing the hard-thresholding operator $\hat{H}_{\boldsymbol{r}}$ in (5.17), thus yielding inexpensive to calculate step sizes. In particular, one example is to neglect the hard-thresholding operator, and instead project the gradient onto the tangent space $\mathbb{T}_{\mathcal{X}_i}\mathcal{M}^{\mathrm{TT}}_{\boldsymbol{r}}(\mathbb{R}^{n_1\times\ldots\times n_d})$ at the current iterate, resulting in

$$\beta_i^{\mathrm{app}} = \underset{\beta\in\mathbb{R}}{\operatorname{argmin}}\left\|\hat{\mathcal{A}}\left(\mathcal{X}_i - \beta\hat{P}_{\mathbb{T}_{\mathcal{X}_i}}(\mathcal{G}(\mathcal{X}_i))\right) - \boldsymbol{b}\right\|_2 = \frac{\left\langle \hat{\mathcal{A}}(\mathcal{X}_i) - \boldsymbol{b} \,,\, \hat{\mathcal{A}}\left(\hat{P}_{\mathbb{T}_{\mathcal{X}_i}}(\mathcal{G}(\mathcal{X}_i))\right)\right\rangle}{\left\|\hat{\mathcal{A}}\left(\hat{P}_{\mathbb{T}_{\mathcal{X}_i}}(\mathcal{G}(\mathcal{X}_i))\right)\right\|_2^2} .$$

For more details on different choices for the step sizes, see the work of Kressner et al. [57] and Rauhut et al. [65].

The advantages of the IHT algorithm are its simplicity and the local recovery guarantees mentioned in the beginning. The main disadvantage is that without further approximation, the complexity of the IHT algorithm is exponential in the order, even in the tensor completion setting. This is due to the use of the TT-SVD in every step, to calculate the low rank approximation of $\mathcal{X}_i - \beta_i\mathcal{G}(\mathcal{X}_i)$. A possible remedy in the completion setting is that the gradient $\mathcal{G}(\mathcal{X}_i) = 2\,\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X}_i) - \boldsymbol{b}\right) = 2\hat{P}_\Omega(\mathcal{X}_i - \mathcal{B})$ is sparse and that the current iterate $\mathcal{X}_i$ is of low rank. As explained in Section 4.4, this could allow the application of the randomized TT-SVD (using structured random tensors) at non-exponential costs. A detailed analysis of the impact of the introduced stochastic error is a subject of future work.

5.3.2 Riemannian Optimization Algorithms

Riemannian optimization algorithms are similar to the IHT algorithm, but more involved. Here, the idea is to use tools from differential geometry to reformulate the restricted optimization problem in $\mathbb{R}^{n_1\times\dots\times n_d}$ (5.16) as an unrestricted optimization problem on the low rank manifold $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$ (see Theorem 3.23). Using the Euclidean metric on $\mathbb{R}^{n_1\times\dots\times n_d}$ induced by the Frobenius scalar product as the metric on the embedded manifold, $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$ becomes a Riemannian manifold. This allows the formulation of different line search algorithms utilizing the Riemannian gradient.

For a target function $J : \mathbb{R}^{n_1\times\dots\times n_d} \to \mathbb{R}$ in the ambient space, let $\tilde{J} : \mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d}) \to \mathbb{R}$ denote its restriction to $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$. Then the Riemannian gradient $\operatorname{Grad}\tilde{J}(\mathcal{X})$ on $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$ can be obtained by projecting the Euclidean gradient onto the tangent space (see e.g. [64, 145]), i.e.

$$\operatorname{Grad}\tilde{J}(\mathcal{X}) = \hat{P}_{\mathbb{T}_{\mathcal{X}}}(\boldsymbol{\nabla} J(\mathcal{X})) \ ,$$

where $\hat{P}_{\mathbb{T}_{\mathcal{X}}}$ is the projector onto the tangent space of $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$ at the point $\mathcal{X}$. Now, given an update direction $\mathcal{Z}$ in the tangent space, e.g. the negative Riemannian gradient, one also needs a way to "move" in that direction on the manifold. In theory, the best method is to use the exponential map $\operatorname{Exp}_{\mathcal{X}} : \mathbb{T}_{\mathcal{X}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d}) \to \mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$, which, starting from $\mathcal{X}$, moves in the direction of $\mathcal{Z}$ along the corresponding geodesic for a distance of $\|\mathcal{Z}\|$. Unfortunately, however, there is no analytic expression for the exponential map available for $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$. Instead, one usually resorts to a *retraction* $R_{\mathcal{X}}$: $\mathbb{T}_{\mathcal{X}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d}) \to \mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$, which is an approximation of the exponential map, see [145] for details. In the tensor train format, an example of a retraction is defined by the TT-SVD via

$$R_{\mathcal{X}}(\mathcal{Z}) := \text{TT-SVD}(\mathcal{X} + \mathcal{Z}) \ ,$$

as shown by Steinlechner [64]. Using these tools, a steepest descent iteration on the manifold $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$ is given by

$$\mathcal{X}_{i+1} := R_{\mathcal{X}_i}\Big(-\beta_i \operatorname{Grad}\tilde{J}(\mathcal{X}_i)\Big) = R_{\mathcal{X}_i}\Big(-\beta_i \hat{P}_{\mathbb{T}_{\mathcal{X}}}(\boldsymbol{\nabla} J(\mathcal{X}_i))\Big) \ , \tag{5.18}$$

where the β_i are some step sizes. Using the TT-SVD as retraction, this is almost equivalent to the IHT algorithm of the previous section, with the distinction, that here the Riemannian gradient is used instead of the Euclidean one.

It is also possible to formulate higher order algorithms, such as the conjugated gradient method in the Riemannian setting. This additionally requires a method of "moving" tangent vectors from the tangent space $\mathbb{T}_{\mathcal{X}_i}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$ at the point $\mathcal{X}_i$ to the tangent space $\mathbb{T}_{\mathcal{X}_{i+1}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\dots\times n_d})$ at a subsequent point $\mathcal{X}_{i+1}$. Again, the optimal differential geometric tool, the *parallel transport*, is computationally infeasible on the TT manifold. However, the *vector transport* introduced by Absil et al. [145] defines a class of approximations, which can be used to accomplish this task.

The differential geometric concepts of retraction and vector transport and their use for optimization on manifolds is introduced in detail in the monograph by Absil et al. [145]. For tensor completion, Riemannian optimization algorithms are formulated and examined by Kressner et al. [57] and Kasai and Mishra [62] in the Tucker and by Steinlechner [63] and Steinlechner [64] in the tensor train format. For the TT-Format, Steinlechner [63] showed that a computational complexity of $\mathcal{O}(d(n + |\Omega|)r^2)$ for one iteration of a Riemannian conjugated gradient algorithms is achievable in the tensor completion setting. In many examples, this allows the reconstruction of a high dimensional tensor in sub-exponential complexity with respect to the order.

5.3.3 Alternating Least Squares

A conceptually different approach is the *alternating least squares* (ALS) algorithm, already described in Section 4.4. Recall that the fundamental idea is to create a nonlinear Gauss-Seidel iteration minimizing the global objective function $J : \mathbb{R}^{n_1 \times \dots \times n_d} \to \mathbb{R}$, by optimizing only one component in the tensor train parametrization at a time. As the parametrization $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d$ is multilinear in the parameters $\mathcal{U}_\mu \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}$, fixing all components $\mathcal{U}_\mu$ except the ν-th component $\mathcal{U}_\nu$ provides a parametrization of $\mathcal{X}$ that is linear in this remaining component,

$$\mathcal{X} := \mathcal{X}(\mathcal{W}_\nu) := \mathcal{U}_1 \circ \ldots \circ \mathcal{W}_\nu \circ \ldots \circ \mathcal{U}_d \qquad \mathcal{W}_\nu \in \mathbb{R}^{r_{\nu-1} \times n_\nu \times r_\nu} .$$

Hence, the global optimization problem in the large ambient space $\mathbb{R}^{n_1 \times \dots \times n_d}$ induces an optimization problem

$$\mathcal{U}_\nu := \operatorname*{argmin}_{\mathcal{W}_\nu \in \mathbb{R}^{r_{\nu-1} \times n_\nu \times r_\nu}} \mathcal{J}(\mathcal{X}(\mathcal{W}_\nu))$$

in the relatively small $\mathbb{R}^{r_{\nu-1} \times n_\nu \times r_\nu}$, which can be solved inexpensively. This procedure is applied sequentially for $\nu = 1, \ldots, d$, which as a whole is called one half-sweep. These half-sweeps are iterated, yielding an algorithm minimizing the global objective function J. For numerical practicability and stability, it is favorable to move the core along and only optimize at the core component. The basic procedure of alternating least squares is outlined in Algorithm 7. For details, the reader is referred to the work of Holtz et al. [97], where further general and implementational details of the ALS can be found. Some general convergence results for ALS-like algorithms are shown by Espig et al. [98].

Using the fixed rank formulation (5.16) of the tensor recovery problem, the global objective function is

$$J(\mathcal{X}) := \left\| \hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b} \right\|_2^2 .$$

For the tensor completion setting, it was shown by Grasedyck et al. [60] that the ALS can be implemented very efficiently for this objective function. In particular, a computational complexity of $\mathcal{O}(dr^4|\Omega|)$ per half-sweep can be realized. For this setting, Grasedyck and Krämer [66] present stable rank adaptation techniques, for the case that the TT-rank is not known a priori.

Algorithm 7: Basic Alternating Least Squares

```
Input  : Initial guess $\mathcal{X} = \mathcal{U}_1, \dots, \mathcal{U}_d \in \mathbb{R}^{n_1 \times \dots \times n_d}$, target function $J : \mathbb{R}^{n_1 \times \dots \times n_d} \to \mathbb{R}$
Output : Solution $\mathcal{X} = \mathcal{U}_1, \dots, \mathcal{U}_d \in \mathbb{R}^{n_1 \times \dots \times n_d}$
1 while stopping criteria not fulfilled do
2     for $\nu = 1, \dots, d$ do
3         Move core to position $\nu$
4         Set $\mathcal{U}_\nu := \underset{\mathcal{W}_\nu \in \mathbb{R}^{r_{\nu-1} \times n_\nu \times r_\nu}}{\operatorname{argmin}} J(\mathcal{U}_1 \circ \dots \circ \mathcal{U}_{\nu-1} \circ W_\nu \circ \mathcal{U}_{\nu+1} \circ \dots \circ \mathcal{U}_d)$
```

5.3.4 Alternating Steepest Descent

In this work, we focus on the alternating steepest descent (ASD) algorithm, which can be considered to lie somewhere in-between the Riemannian algorithms and the ALS. The fundamental idea is to use the closed form of the projection operator $\mathbb{T}_{\mathcal{X}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ onto the tangent space of the TT-manifold, established in Theorem 3.25, to define a gradient descent algorithm optimizing in only one of the spaces V_μ from Theorem 3.25 at a time. The ASD algorithm can be seen as a generalization of the alternating directional fitting (ADF) algorithm developed by Grasedyck et al. [59, 60] for tensor completion. However, the derivation as a projected gradient descent presented in the following is quite different and provides additional insight into the algorithm.

As established in Theorem 3.23, the set of TT-rank $\boldsymbol{r}$ tensors $\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ forms a smooth embedded manifold and according to Theorem 3.24, given a point $\mathcal{X} = \mathcal{U}_1 \circ \dots \circ \mathcal{U}_d \in \mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ with left-orthogonal $\mathcal{U}_1, \dots, \mathcal{U}_{d-1}$, the tangent space $\mathbb{T}_{\mathcal{X}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1 \times \dots \times n_d})$ at $\mathcal{X}$ can be expressed as

$$\mathbb{T}_{\mathcal{X}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1 \times \dots \times n_d}) = V_1 \oplus \dots \oplus V_d \ ,$$

where

$$\begin{aligned} V_1 &= \{\mathcal{K} \circ \mathcal{U}_2 \circ \dots \circ \mathcal{U}_d \mid \mathcal{K} \in \mathbb{R}^{n_1 \times r_1},\ \mathcal{U}_1 \circ \mathcal{K} = \mathbf{0}\} \ , \\ V_\mu &= \left\{\mathcal{U}_1 \circ \dots \circ \mathcal{U}_{\mu-1} \circ \mathcal{K} \circ \mathcal{U}_{\mu+1} \circ \dots \circ \mathcal{U}_d \mid \mathcal{K} \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu},\ \mathcal{U}_\mu *_{(1,2),(1,2)} \mathcal{K} = \mathbf{0}\right\} \ , \end{aligned}$$

for $\mu = 1, \dots, d-1$ and

$$V_d = \{\mathcal{U}_1 \circ \dots \circ \mathcal{U}_{d-1} \circ \mathcal{K} \mid \mathcal{K} \in \mathbb{R}^{r_{d-1} \times n_d}\} \ .$$

The essential idea of the ASD algorithm is to create an iteration by projecting the Euclidean gradient of some target functional $J : \mathbb{R}^{n_1 \times \dots \times n_d} \to \mathbb{R}$ onto *one* of the spaces V_μ at a time and perform an update in that direction. For this purpose, it is required that the spaces partition the tangent space, but not necessarily that they are orthogonal or disjoint. We can therefore simplify the calculation and in practice accelerate the algorithm, by considering the enlarged spaces

$$\begin{aligned}
\tilde{V}_1 &= \{\mathcal{K} \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_d \mid \mathcal{K} \in \mathbb{R}^{n_1 \times r_1}\} \ , \\
\tilde{V}_\mu &= \{\mathcal{U}_1 \circ \ldots \circ \mathcal{U}_{\mu-1} \circ \mathcal{K} \circ \mathcal{U}_{\mu+1} \circ \ldots \circ \mathcal{U}_d \mid \mathcal{K} \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}\} \ , \\
\tilde{V}_d &= \{\mathcal{U}_1 \circ \ldots \circ \mathcal{U}_{d-1} \circ \mathcal{K} \mid \mathcal{K} \in \mathbb{R}^{r_{d-1} \times n_d}\} \ ,
\end{aligned} \tag{5.19}$$

where we drop the orthogonality constraint for $1 \leq \mu < d$. Next, we show that these enlarged spaces span the tangent space and nothing more. In order to avoid unnecessary clutter, we will omit the special treatment for $\mu = 1$ and $\mu = d$ in the following, whenever the necessary changes to the formulas are self-evident.

Proposition 5.17. *For the enlarged spaces $\tilde{V}_\mu$ from* (5.19)*, it holds that*

$$\mathbb{T}_{\mathcal{X}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}\left(\mathbb{R}^{n_1 \times \ldots \times n_d}\right) = \tilde{V}_1 + \ldots + \tilde{V}_d \ .$$

Proof. Let $\mathcal{I}_{r_{\mu-1} \times n_\mu} \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_{\mu-1} \times n_\mu}$ denote the generalized identity, in the sense that $\boldsymbol{I}^{<2>}_{r_{\mu-1} \times n_\mu} = \boldsymbol{I} \in \mathbb{R}^{(r_{\mu-1} n_\mu) \times (r_{\mu-1} n_\mu)}$. Note that $\mathcal{U}_\mu$ is left-orthogonal for all $\mu < d$. Therefore, for any $\mathcal{K} \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}$ there holds

$$\begin{aligned}
\mathcal{K} &= \underbrace{\left(\mathcal{I}_{r_{\mu-1} \times n_\mu} - (\mathcal{U}_\mu *_{3,3} \mathcal{U}_\mu)\right) *_{(1,2),(1,2)} \mathcal{K}}_{=:\mathcal{K}_\perp \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}} + \underbrace{(\mathcal{U}_\mu *_{3,3} \mathcal{U}_\mu) *_{(1,2),(1,2)} \mathcal{K}}_{=:\mathcal{K}_\| \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}} \\
&= \underbrace{\left(\mathcal{I}_{r_{\mu-1} \times n_\mu} - (\mathcal{U}_\mu *_{3,3} \mathcal{U}_\mu)\right) *_{(1,2),(1,2)} \mathcal{K}}_{=:\mathcal{K}_\perp \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}} + \mathcal{U}_\mu \circ \underbrace{\left(\mathcal{U}_\mu *_{(1,2),(1,2)} \mathcal{K}\right)}_{=:\tilde{\mathcal{K}}_\| \in \mathbb{R}^{r_\mu \times r_\mu}}
\end{aligned}$$

and it is easy to see that $\mathcal{U}_\mu *_{(1,2),(1,2)} \mathcal{K}_\perp = \mathbf{0}$. From this, it follows that

$$\begin{aligned}
&\mathcal{U}_1 \circ \ldots \circ \mathcal{U}_{\mu-1} \circ \mathcal{K} \circ \mathcal{U}_{\mu+1} \circ \ldots \circ \mathcal{U}_d \\
=&\underbrace{\mathcal{U}_1 \circ \ldots \circ \mathcal{U}_{\mu-1} \circ \mathcal{K}_\perp \circ \mathcal{U}_{\mu+1} \circ \ldots \circ \mathcal{U}_d}_{\in V_\mu} + \underbrace{\mathcal{U}_1 \circ \ldots \circ \mathcal{U}_{\mu-1} \circ \mathcal{U}_\mu \circ \left(\tilde{\mathcal{K}}_\| \circ \mathcal{U}_{\mu+1}\right) \circ \ldots \circ \mathcal{U}_d}_{\in \tilde{V}_{\mu+1}}
\end{aligned}$$

holds, i.e. any element of $\tilde{V}_\mu$ can be decomposed as one element from V_μ and one from $\tilde{V}_{\mu+1}$. As $\tilde{V}_d = V_d$ holds, this implies that the $\tilde{V}_\mu$ and V_μ span the same space, which concludes the proof. □

Note, however, that in contrast to the V_μ, the enlarged spaces $\tilde{V}_\mu$ are neither disjoint nor orthogonal anymore. Using the closed form of the tangent space projection from Theorem 3.25, the projectors $\hat{P}_\mu : \mathbb{R}^{n_1 \times \ldots \times n_d} \to \tilde{V}_\mu \subseteq \mathbb{T}_{\mathcal{X}}\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1 \times \ldots \times n_d})$ (called $\hat{P}_\mu^+$ in the theorem) onto the enlarged spaces $\tilde{V}_\mu$ can be given as

$$\hat{P}_\mu(\mathcal{Y}) = \left(\mathcal{U}_{<\mu} *_{(\mu),(\mu)} \mathcal{U}_{<\mu}\right) \circ_{\mu-1} \mathcal{Y} \circ_{d-\mu} \left(\mathcal{U}_{>\mu} *_{(1),(1)} \mathcal{U}_{>\mu}\right) \ ,$$

where we used the tensors $\mathcal{U}_{<\mu}$ and $\mathcal{U}_{>\mu}$ from Theorem 3.25. In the alternating steepest descent, we use the negative Euclidean gradient $\nabla J(\mathcal{X}_i)$ at the current iterate as update direction. An abstract outline of the resulting procedure is provided in Algorithm 8.

Algorithm 8: Abstract Alternating Steepest Descent

```
Input  : Initial guess $\mathcal{X} \in \mathbb{R}^{n_1\times\ldots\times n_d}$, target function $J : \mathbb{R}^{n_1\times\ldots\times n_d} \to \mathbb{R}$
Output : Solution $\mathcal{X} \in \mathbb{R}^{n_1\times\ldots\times n_d}$
1 for $i = 1, 2, \ldots$ do
2     for $\mu = 1, \ldots, d$ do
3         Calculate the projected gradient $\hat{P}_\mu(\nabla J(\mathcal{X}))$
4         Determine step size $\beta_i$
5         Update $\mathcal{X} := \mathcal{X} - \beta_i \hat{P}_\mu(\nabla J(\mathcal{X}))$
```

In some respects, this is similar to the Riemannian gradient descent in (5.18), with the main difference that instead of using the complete Riemannian gradient in each iteration, its orthogonal projection onto one of the spaces $\tilde{V}_\mu$ is used. The main advantage is that independent of step size, such an update never leaves the manifold and therefore, no retraction is required at all.[4]

Lemma 5.18. *Let $\mathcal{X} \in \mathbb{R}^{n_1\times\ldots\times n_d}$ be a tensor of rank $\boldsymbol{r}$ given in a TT-decomposition $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d$ with core position μ. Given an update direction $\mathcal{Z} \in \mathbb{R}^{n_1\times\ldots\times n_d}$, the update $\mathcal{X}_+ = \mathcal{X} + \beta\hat{P}_\mu(\mathcal{Z})$ can be given as*

$$\mathcal{X}_+ = \mathcal{U}_1 \circ \ldots \mathcal{U}_{\mu-1} \circ (\mathcal{U}_\mu + \beta\mathcal{W}_\mu) \circ \mathcal{U}_{\mu+1} \circ \ldots \circ \mathcal{U}_d = \mathcal{U}_{<\mu} \circ (\mathcal{U}_\mu + \beta\mathcal{W}_\mu) \circ \mathcal{U}_{>\mu} \ ,$$

with

$$\mathcal{W}_\mu := \mathcal{U}_{<\mu} *_{(1,\ldots,\mu-1),(1,\ldots,\mu-1)} (\mathcal{Z}) *_{(\mu+1,\ldots,d),(2,\ldots,d-\mu+1)} \mathcal{U}_{>\mu} \quad \in \mathbb{R}^{r_{\mu-1}\times n_\mu \times r_\mu} \ . \tag{5.20}$$

In particular, the rank of $\mathcal{X}_+$ is bounded by the rank of $\mathcal{X}$.

Proof. The result follows directly from the observation that

$$\begin{aligned}\hat{P}_\mu(\mathcal{Z}) &= \Big(\mathcal{U}_{<\mu} *_{(\mu),(\mu)} \mathcal{U}_{<\mu}\Big) \circ_{\mu-1} (\mathcal{Z}) \circ_{d-\mu} \Big(\mathcal{U}_{>\mu} *_{(1),(1)} \mathcal{U}_{>\mu}\Big) \\ &= \mathcal{U}_{<\mu} \circ \underbrace{\Big(\mathcal{U}_{<\mu} *_{(1,\ldots,\mu-1),(1,\ldots,\mu-1)} (\mathcal{Z}) *_{(\mu+1,\ldots,d),(2,\ldots,d-\mu+1)} \mathcal{U}_{>\mu}\Big)}_{=\mathcal{W}_\mu} \circ \mathcal{U}_{>\mu}\end{aligned}$$

holds and the linearity of the contraction. □

In contrast to the Riemannian optimization algorithms as introduced in Section 5.3.2, which rely on a retraction to stay on the manifold, this allows a direct calculation of the optimal step size with respect to the global target function J. Given the current iterate $\mathcal{X}$ and an update direction $\mathcal{Z}$, the optimal step size is given by

$$\beta = \underset{\beta\in\mathbb{R}}{\operatorname{argmin}} \Big\| \hat{\mathcal{A}}\Big(\mathcal{X} - \beta\hat{P}_\mu(\mathcal{Z})\Big) - \boldsymbol{b} \Big\|_2^2 = \frac{\Big\langle \hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b},\ \hat{\mathcal{A}}(\hat{P}_\mu(\mathcal{Z}))\Big\rangle}{\Big\|\hat{\mathcal{A}}(\hat{P}_\mu(\mathcal{Z}))\Big\|_2^2} \ .$$

[4]That is, the rank cannot increase. Theoretically, the update could end in a critical point, i.e. a tensor with smaller rank.

If the update is in direction of the gradient $\mathcal{Z} = \hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)$, we can use the definition of $\mathcal{W}_\mu$ from (5.20) and simplify this to

$$\beta = \frac{\Big\langle \hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b},\ \hat{\mathcal{A}}\Big(\hat{P}_\mu\Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big)\Big)\Big\rangle}{\Big\|\hat{\mathcal{A}}\Big(\hat{P}_\mu\Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big)\Big)\Big\|_2^2} = \frac{\|\mathcal{W}_\mu\|_F^2}{\Big\|\hat{\mathcal{A}}\Big(\hat{P}_\mu\Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big)\Big)\Big\|_2^2} ,$$

where we used that $\hat{P}_\mu$ is an orthogonal subspace projector and the fact that the complete Frobenius norm of a tensor in TT-representation can be obtained from the core component.

As a further side effect, Proposition 5.18 shows that it suffices to calculate (5.20) and then perform a computationally inexpensive addition exclusively within the core component. Using a different derivation, Grasedyck et al. [60] show that this can be exploited to create a very efficient algorithm for the tensor completion problem, exhibiting a computational complexity of only $\mathcal{O}(dr^2|\Omega|)$ per half-sweep.[5] Here, half-sweep means one step for each projector $\hat{P}_\mu, 1 \le \mu \le d$. In the following, we show that a similar construction allows an efficient algorithm for the more general setting of tensor recovery using rank-one samples, as defined in (5.7).

Consider the optimization problem in (5.16), with a measurement operator

$$\hat{\mathcal{A}}(\mathcal{X})[k] := \sum_{i_1,\dots,i_d} \boldsymbol{a}_{1,k}[i_1]\ \boldsymbol{a}_{2,k}[i_2] \dots \boldsymbol{a}_{d,k}[i_d] \mathcal{X}[i_1,\dots,i_d] \qquad\qquad \boldsymbol{a}_{\mu,k} \in \mathbb{R}^{n_\mu}$$

using N rank-one samples. At the heart of an efficient implementation are stacks $\mathcal{S}^{\text{Left}}_{\mu,k}$ and $\mathcal{S}^{\text{Right}}_{\mu,k}$ defined as

$$\begin{aligned}
\mathcal{S}^{\text{Left}}_{1,k} &:= \boldsymbol{a}_{1,k} \circ \mathcal{U}_1 && \in \mathbb{R}^{r_1} \\
\mathcal{S}^{\text{Left}}_{\mu,k} &:= \boldsymbol{a}_{\mu,k} \circ \Big(\mathcal{S}^{\text{Left}}_{\mu-1,k} \circ \mathcal{U}_\mu\Big) && \in \mathbb{R}^{r_\mu} && (5.21)\\
\mathcal{S}^{\text{Right}}_{d,k} &:= \mathcal{U}_d \circ \boldsymbol{a}_{d,k} && \in \mathbb{R}^{r_{d-1}} \\
\mathcal{S}^{\text{Right}}_{\mu,k} &:= \Big(\mathcal{U}_\mu \circ \mathcal{S}^{\text{Right}}_{\mu+1,k}\Big) \circ \boldsymbol{a}_{\mu,k} && \in \mathbb{R}^{r_{\mu-1}} , && (5.22)
\end{aligned}$$

where the $\mathcal{U}_\mu$ in the definition of $\mathcal{S}^{\text{Left}}_{\mu,k}$ are assumed to be left-orthogonal and those in the definition of $\mathcal{S}^{\text{Right}}_{\mu,k}$ are assumed to be right-orthogonal. The calculations of $\mathcal{S}^{\text{Left}}_{\mu,k}$ and $\mathcal{S}^{\text{Right}}_{\mu,k}$ each have computational complexity $\mathcal{O}(nr^2)$, assuming that $\mathcal{S}^{\text{Left}}_{\mu-1,k}$ and $\mathcal{S}^{\text{Right}}_{\mu+1,k}$, as well as the appropriately orthogonalized $\mathcal{U}_\mu$, are directly available. The crucial components of the algorithm are $\hat{\mathcal{A}}(\hat{P}_\mu(\nabla J(\mathcal{X}))) = \hat{\mathcal{A}}(\hat{P}_\mu(\hat{\mathcal{A}}^T(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b})))$, which are needed for the calculation of the optimal step size and the core component of projected update direction

$$\mathcal{W}_\mu := \mathcal{U}_{<\mu} *_{(1,\dots,\mu-1),(1,\dots,\mu-1)} \Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big) *_{(\mu+1,\dots,d),(2,\dots,d-\mu+1)} \mathcal{U}_{>\mu} .$$

[5]Note that the ADF algorithm of Grasedyck et al. [60] is not completely equivalent to the ASD algorithm presented here. From the point of view of this work, the main difference is that the ADF further divides the tangent space using the external dimension. In the tensor completion setting, this allows a fine grained calculation of the step sizes. However, this does not extend to the more general tensor recovery setting.

Using the stacks, these crucial components can be calculated inexpensively, as we have

$$\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)[k] = \mathcal{S}^{\text{Left}}_{\mu-1,k} \circ (\boldsymbol{a}_{\mu,k} *_{1,2} \mathcal{U}_\mu) \circ \mathcal{S}^{\text{Right}}_{\mu,k} - \boldsymbol{b}[k] \tag{5.23}$$

$$\mathcal{W}_\mu = \sum_{k=1}^{N} \Big(\hat{\mathcal{A}}(\mathcal{X}_i) - \boldsymbol{b}\Big)[k] \; \Big(\mathcal{S}^{\text{Left}}_{\mu,k} \otimes \boldsymbol{a}_{\mu,k} \otimes \mathcal{S}^{\text{Right}}_{\mu,k}\Big) \tag{5.24}$$

$$\hat{\mathcal{A}}\Big(\hat{P}_\mu\Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big)\Big)[k] = \mathcal{S}^{\text{Left}}_{\mu,k} \circ (\boldsymbol{a}_{\mu,k} *_{1,2} \mathcal{W}_\mu) \circ \mathcal{S}^{\text{Right}}_{\mu,k} \;, \tag{5.25}$$

with the obvious exception that for the special cases $\mu = 1$ and $\mu = d$, the left/right stack is omitted, respectively. The computational complexity for each of the above calculations is $\mathcal{O}(nr^2N)$. Therefore, the complete procedure for tensor recovery via rank one samples, given in Algorithm 9, has complexity $\mathcal{O}(dnr^2N + dnr^3)$ for one sweep. As the number of samples N has to exceed the degrees of freedoms $\Theta(dnr^2)$ for successful reconstruction, the computational cost for the core movement is usually negligible, resulting in a computational complexity of $\mathcal{O}(dnr^2N)$ per half-sweep.

5.3.5 Block Alternating Steepest Descent

Just as most of the algorithms presented in this chapter, the alternating steepest descent algorithm optimizes on the manifold of tensors with a fixed TT-rank. This means that either an initial guess of the optimal rank has to be known a priori, or rank adaptation strategies are required. The first approach is naturally very efficient, if a sufficiently accurate rank estimate is available, however, for many settings this is hard to obtain. The simplest rank adaptation strategy is to increase the TT-ranks by adding a rank-increasing constant or random perturbation to the current iterate, whenever the manifold optimization reaches a (local) minimum. However, this has the disadvantage that not a specific rank is increased, but rather all ranks are increased at once. More elaborate but also more expensive approaches estimate the potential gain of a rank increase at each position or increase the rank by adding a low rank truncation of the Euclidean gradient. In a recent work, Grasedyck and Krämer [66] introduce further rank adaptation techniques for tensor completion via the ALS ensuring stability of the reconstruction.

In this section, we propose a new rank adaptation strategy for tensor recovery via the ASD algorithm based on the block-TT format. This strategy allows for the selective increase and decrease of ranks at beneficial positions and also provides a specific subspace for the increase. The main tool is the block-TT format, which was first introduced by Dolgov et al. [146] for the calculation of extreme eigenvalues and allows the representation of several tensors in the TT-format, which share the same orthogonal basis for all but one mode.

Definition 5.19 (Block Tensor Train [146])**.** The tensors $\mathcal{X}_1, \dots, \mathcal{X}_L \in \mathbb{R}^{n_1\times\dots\times n_d}$ are said to be in block-TT format with rank $\boldsymbol{r} = (r_1, \dots, r_{d-1})$ and core position μ, if there exist left orthogonal $\mathcal{U}_1 \in \mathbb{R}^{n_1\times r_1}, \dots, U_{\mu-1} \in \mathbb{R}^{r_{\mu-2}\times n_{\mu-1}\times r_{\mu-1}}$, right orthogonal $\mathcal{U}_{\mu+1} \in \mathbb{R}^{r_\mu\times n_{\mu+1}\times r_{\mu+1}}, \dots, \mathcal{U}_d \in \mathbb{R}^{r_{d-1}\times n_d}$ and $\mathcal{U}_{\mu,1}, \dots, \mathcal{U}_{\mu,L} \in \mathbb{R}^{r_{\mu-1}\times n_\mu\times r_\mu}$, such that for

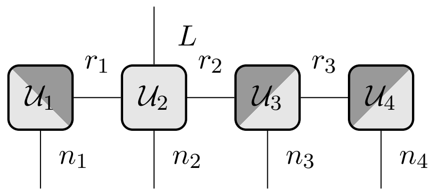
Algorithm 9: Alternating Steepest Descent for Tensor Recovery

```
Input: Initial guess $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d \in \mathbb{R}^{n_1 \times \ldots \times n_d}$, measurement operator
       $\hat{\mathcal{A}} : \mathbb{R}^{n_1 \times \ldots \times n_d} \to \mathbb{R}^N$ and measured values $\boldsymbol{b} \in \mathbb{R}^N$
Output: Solution $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d \in \mathbb{R}^{n_1 \times \ldots \times n_d}$
1  Subroutine Update_core($\mu$)
2      Calculate $\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}$ according to (5.23).
3      Calculate $\mathcal{W}_\mu$ according to (5.24).
4      Calculate $\hat{\mathcal{A}}(\hat{P}_\mu(\hat{\mathcal{A}}^T(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b})))$ according to (5.25).
5      Caclulate step size as $\beta = \frac{\|\mathcal{W}_\mu\|_F^2}{\|\hat{\mathcal{A}}(\hat{P}_\mu(\hat{\mathcal{A}}^T(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b})))\|_2^2}$
6      Update $\mathcal{U}_\mu := \mathcal{U}_\mu - \beta \mathcal{W}_\mu$.
7  begin
8      Move core to position 1.
9      Calculate $\mathcal{S}^{\text{Right}}_{\mu,k}$ for $\mu = d, \ldots, 2$, according to (5.22).
10     while stopping criteria not fullfilled do
11         for $\mu = 1, \ldots, d$ do
12             Update_core($\mu$)
13             if $\mu < d$ then
14                 Move core to position $\mu + 1$
15                 Calculate $\mathcal{S}^{\text{Left}}_{\mu,k}$ according to (5.21).
16         for $\mu = d, \ldots, 1$ do
17             Update_core($\mu$)
18             if $\mu > 1$ then
19                 Move core to position $\mu - 1$
20                 Calculate $\mathcal{S}^{\text{Right}}_{\mu,k}$ according to (5.22).
```

each $l = 1, \ldots, L$, it holds that

$$\mathcal{X}_l = \mathcal{U}_1 \circ \ldots \mathcal{U}_{\mu-1} \circ \mathcal{U}_{\mu,l} \circ \mathcal{U}_{\mu+1} \circ \ldots \circ \mathcal{U}_d \ .$$

This means that $\mathcal{X}_1, \ldots, \mathcal{X}_L$ can be represented in the same TT-representation, with only the core tensor changing. It will often be convenient to use a core tensor $\mathcal{U}_\mu \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times L \times r_\mu}$ of order four, where the third mode encodes the additional index. In the following, an example of this case is shown in diagrammatic notation.



In contrast to the classic TT-format, moving the core position in the block TT-format can in general increase the rank, because the extra index is moved along with the core. For example, to move the core from position μ to $\mu + 1$ one can use a SVD and calculate

$$\begin{aligned} \boldsymbol{V}^{<2>}\boldsymbol{\Sigma}\boldsymbol{W}^{<1>} &:= \mathrm{SVD}\Big(\boldsymbol{U}_\mu^{<2>}\Big) \qquad \mathcal{V} \in \mathbb{R}^{r_{\mu-1}\times n_m u \times r}, \boldsymbol{\Sigma} \in \mathbb{R}^{r\times r}, \mathcal{W} \in \mathbb{R}^{r\times L\times r_\mu} \\ \mathcal{U}_\mu^{<2>} &:= \mathcal{V} \\ \mathcal{U}_{\mu+1} &:= \mathrm{reshape}_{(1,3,2,4)}(\boldsymbol{\Sigma} \circ \mathcal{W} \circ \mathcal{U}_{\mu+1}) \qquad \in \mathbb{R}^{r\times n_{\mu+1}\times L\times r_\mu} \ . \end{aligned}$$

The rank of $\boldsymbol{U}_\mu^{<2>} \in \mathbb{R}^{r_{\mu-1}n_\mu \times L r_\mu}$, which becomes the new rank r_μ, is in general between one and $\min(r_{\mu-1}n_\mu,\ Lr_\mu)$, i.e. possibly larger than r_μ. Analogously, it is possible to move the core from μ to $\mu - 1$ and by repetition also to any other position. Note that using an SVD with a specified accuracy ϵ, it is therefore possible that the rank of the block tensor train increases or decreases when moving the core. In the following, we show how this can be used to formulate a rank adaptive algorithm.

In the proposed block alternating steepest descent, we use the block-TT format to perform several separate ASD iterations in parallel, while enforcing a coherent basis for all components which are not currently optimized. To this end, the set of measurement is randomly split into several subsets. This gives rise to independent measurement operators $\hat{\mathcal{A}}_l$, right hand sides $\boldsymbol{b}_l$ and thereby to independent optimization problems

$$\begin{aligned} \text{minimize } & J_l(\mathcal{X}_l) := \|\mathcal{A}_l(\mathcal{X}) - \boldsymbol{b}_l\|_2 \\ \text{subject to } & \mathcal{X}_l \in \mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\ldots\times n_d}) \ . \end{aligned}$$

For each of these optimization problems, the ASD is used, with the addition that when moving the core after each micro iteration, the block TT-format is preserved, i.e. the core is moved using a complete SVD with prescribed accuracy ϵ, as described above. The heuristic is that if the basis of the true solution is found, all iterates stay within this subspace and moving the core will not increase the rank. However, if the current basis is too small, the different sub-problems evolve in different directions, which can be quantified by the singular value decomposition and is used subsequently to increase the rank when moving the core. As this is done separately for each core position, this allows a selective rank increase and also the prescription of an accuracy, at which a rank increase or decrease should be carried out. The current solution can always be obtained by averaging over all of the parallel iterates. However, note that since each sub-problem uses only a subset of the measurements, the residual of this iteration is not necessarily non-increasing in every step. It is therefore expedient to keep track of the best solution separately. In practice, there is also the problem of over-fitting the data, e.g. in the presence of noise. To prevent this, we use a control set containing a small amount of the measurements, which are never used in any of the L normal sets. If the error of the solution on the control set fails to decrease during the iteration, this is an indication of over-fitting, which can for example be used to increase the ϵ, i.e. decrease the accuracy when moving the core. The complete algorithm is given in Algorithm 10.

Surprisingly, the rank adaptation method via the block-TT format can be added without significant increase of the computational complexity compared to the standard ASD algorithm. In particular, the computational complexity of Algorithm 10 for one half-sweep scales as $\mathcal{O}(dnr^2N + dnr^3L^2)$, i.e. for the usual case that $rL^2 < N$, the complexity is not increased.

Algorithm 10: Block Alternating Steepest Descent for Tensor Recovery

```
Input: Initial guess $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d \in \mathbb{R}^{n_1 \times \ldots \times n_d}$, measurement operator
       $\hat{\mathcal{A}} : \mathbb{R}^{n_1 \times \ldots \times n_d} \to \mathbb{R}^N$ and measured values $\boldsymbol{b} \in \mathbb{R}^N$, division constant $L \in \mathbb{N}_+$,
       accuracy $\epsilon > 0$
Output: Solution $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d \in \mathbb{R}^{n_1 \times \ldots \times n_d}$
1  Subroutine Update_core($\mu$)
2    for $l = 1, \ldots, L$ do
3      Using only the samples $k \in \Lambda_l$:
4      Calculate $\hat{\mathcal{A}}_l(\mathcal{X}_l) - \boldsymbol{b}_l$ according to (5.23).
5      Calculate $\mathcal{W}_{\mu,l}$ according to (5.24).
6      Calculate $\hat{\mathcal{A}}_l(\hat{P}_\mu(\hat{\mathcal{A}}_l^T(\hat{\mathcal{A}}_l(\mathcal{X}_l) - \boldsymbol{b}_l)))$ according to (5.25).
7      Calculate step size as $\beta = \frac{\|\mathcal{W}_{\mu,l}\|_F^2}{\|\hat{\mathcal{A}}_l(\hat{P}_\mu(\hat{\mathcal{A}}_l^T(\hat{\mathcal{A}}_l(\mathcal{X}_l) - \boldsymbol{b}_l)))\|_2^2}$
8      Update $\mathcal{U}_\mu[i_1, i_2, l, i_3] := \mathcal{U}_\mu[i_1, i_2, l, i_3] - \beta \mathcal{W}_{\mu,l}[i_1, i_2, i_3]$.
9  begin
10   Set $R_{\text{best}} = \infty$.
11   Move core to position 1.
12   Duplicate core $L$ times: $\mathcal{U}_1[i_1, i_2, l, i_3] := \mathcal{U}_1[i_1, i_2, i_3] \quad \forall i_1, i_2, i_3, l$.
13   Separate samples $\{1, \ldots, N\}$ into the optimization set $\Lambda$ and control set $\Lambda_{\text{test}}$.
14   Calculate $\mathcal{S}_{\mu,k}^{\text{Right}}$ for $\mu = d, \ldots, 2$, according to (5.22).
15   while stopping criteria not fullfilled do
16     Partition the samples into sets, $\Lambda_l \subseteq \Lambda$ for $1 \leq l \leq L$.
17     for $\mu = 1, \ldots, d$ do
18       Update_core($\mu$)
19       if $\mu < d$ then
20         Move core to position $\mu + 1$, with accuracy $\epsilon$
21         Calculate $\mathcal{S}_{\mu,k}^{\text{Left}}$ according to (5.21).
22     for $\mu = d, \ldots, 1$ do
23       Update_core($\mu$)
24       if $\mu > 1$ then
25         Move core to position $\mu - 1$, with accuracy $\epsilon$
26         Calculate $\mathcal{S}_{\mu,k}^{\text{Right}}$ according to (5.22).
27     Calculate average core $\mathcal{U}_{\text{avg}}[i_1, i_2, i_3] = \frac{1}{L} \sum_{l=1}^{L} \mathcal{U}_1[i_1, i_2, l, i_3]$.
28     Calculate residual $R_{\text{test}}$ on $\Lambda_{\text{test}}$ using $\mathcal{U}_{\text{avg}} \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_d$.
29     if $R_{test} < R_{best}$ then
30       Set $R_{\text{best}} = R_{\text{test}}$.
31       Set $\mathcal{X}_{\text{best}} = \mathcal{U}_{\text{avg}} \circ \mathcal{U}_2 \circ \ldots \circ \mathcal{U}_d$.
32   return $\mathcal{X}_{\text{best}}$.
```

5.4 Numerical Experiments

This section presents several experiments providing numerical evidence that low rank tensor recovery using a limited number of rank-one samples or incomplete sets of entries is indeed possible and numerically feasible. In all experiments, we use the (block) alternating steepest descent algorithm introduced in the previous section.

Analogously to Section 4.5, the experiments use different types of random tensors and random measurements, which are created as follows.

- **Standard Gaussian tensors** are created by sampling each entry independently from $\mathcal{N}(0,1)$.
- **Random rank $\boldsymbol{r}$ tensors** are created by sampling the entries of the components $\mathcal{U}_1, \dots, \mathcal{U}_d$ in representation (3.12) independently from $\mathcal{N}(0,1)$, i.e. all components $\mathcal{U}_\mu$ are independent standard Gaussian random tensors.
- **Random tensors with roughly quadratically decaying singular values** are created from random low rank tensors by imposing the specific decay on the singular values of all relevant matricizations. To this end, for μ from 1 to $d-1$, the components $\mathcal{U}_\mu \circ \mathcal{U}_{\mu+1}$ are contracted and re-separated by calculating the SVD $\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{V}^T = \mathrm{Mat}_{(1,2)}(\mathcal{U}_\mu \circ \mathcal{U}_{\mu+1})$. The $\boldsymbol{\Sigma}$ factor is then replaced with a matrix $\tilde{\boldsymbol{\Sigma}}$, in which the singular values decay in the desired way. For a quadratic decay that is $\tilde{\boldsymbol{\Sigma}} := \mathrm{diag}(1, \frac{1}{2^2}, \frac{1}{3^2}, \dots, \frac{1}{100^2}, 0, \dots, 0)$, where 100 is a cut-off used in all of the following experiments. Subsequently, $\boldsymbol{U}_\mu^{<2>} = \boldsymbol{W}$ and $\boldsymbol{U}_{\mu+1}^{<1>} = \tilde{\boldsymbol{\Sigma}}\boldsymbol{V}^T$ are replaced. Note that since the later steps change the singular values of the earlier matricization, the singular values of the resulting tensor do *not* obey the desired decay exactly, see also the notes in Section 4.5.
- **Random rank one samples** are created by sampling the vectors $\boldsymbol{a}_{\mu,k}$ in (5.7) as independent standard Gaussian tensors (vectors), i.e. each entry is sampled independently from $\mathcal{N}(0,1)$. For exact measurements, the measured values are by definition given as $\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T})$, where $\mathcal{T}$ is the target tensor. For noisy measurements, a standard Gaussian vector $\boldsymbol{g} \in \mathbb{R}^N$ is created and the measured values are set to

$$\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T}) + \tau \frac{\left\|\hat{\mathcal{A}}(\mathcal{T})\right\|_2}{\|\boldsymbol{g}\|_2} \boldsymbol{g} \ , \tag{5.26}$$

  where $\tau > 0$ is the noise amplitude.
- **Random completion samples** are created by randomly sampling N positions of the tensor. Each position is drawn from a uniform distribution on all remaining (i.e. not already drawn) positions. In the noiseless setting, the measured values are exactly the entries at the sampled positions, i.e. $\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T})$. Noisy measurements are

created the same way as for random rank one samples, i.e.

$$\boldsymbol{b} = \hat{\mathcal{A}}(\mathcal{T}) + \tau \frac{\left\|\hat{\mathcal{A}}(\mathcal{T})\right\|_2}{\|\boldsymbol{g}\|_2}\, \boldsymbol{g} \ , \tag{5.27}$$

where $\tau > 0$ is the noise amplitude and $\boldsymbol{g} \in \mathbb{R}^N$ a standard Gaussian vector.

All calculations for this section are performed using the `xerus` C++ tensor library (see Appendix A), which includes the complete implementation of the (block) ASD algorithm for tensor recovery. The following implementation details and additions to the base algorithm are used. For the block ASD, we use $L = 2$ equally large sample sets plus the control set, which contains 10% of the measurements. We update the current "best solution", if for the current iterate the residual norm on the control set is at least by a factor 0.99 smaller than for the current best solution. Both the block and normal ASD algorithm are stopped, if at the end of one iteration either the current residual norm is smaller than 10^{-10}, or 1000 iterations were performed. Additionally, the fixed rank ASD algorithm is stopped if the residual norm did not improve by at least a factor 0.999 during the last ten iterations.[6] For the block ASD, the residual norm is not necessarily decreasing, which is why we use the control set here. In particular, the block ASD is stopped if for 25 subsequent iterations, the best solution is not updated. At the beginning of the iteration, the accuracy is $\epsilon = 5 \cdot 10^{-3}$, which is increased by a factor two (to a maximum of 0.32) if no new best solution was found in the previous three iterations. In all experiments, we restrict each entry of TT-rank to remain less than or equal to 20 during the iteration.

Unless explicitly stated otherwise, all experiments use uniform dimensions $n_\mu = n$ and target ranks $r_\mu = r$. In most experiments, the following values of interest depending on the solution (or iterate) $\mathcal{X}$ are calculated. The *relative reconstruction error* is given as

$$\frac{\|\mathcal{X} - \mathcal{T}\|}{\|\mathcal{T}\|}$$

and the *relative residual norm* is given as

$$\frac{\left\|\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\right\|}{\|\boldsymbol{b}\|} \ .$$

In most experiments, we calculate the averages of these values over 200 independent runs for each parameter combination. In the calculation of these averages, we cap both the relative reconstruction error, as well as the relative residual at one, which represents a failed reconstruction. In the noiseless settings, we say that the reconstruction is successful if the relative reconstruction error is smaller than 10^{-5}. In all experiments, we observe that if the algorithm reaches this threshold, there is a linear convergence of the error down to machine precision. In the noisy settings, we say that the reconstruction is successful if the reconstruction is denoising, i.e. the reconstruction error is smaller then the noise amplitude.

[6]In fact, the factor is $0.9999^{10} \approx 0.999$.

Note that for all experiments presented in the following, there are exhaustive numerical results available in Appendix B, including the ratio of successful reconstructions, the average relative residual norm, the average relative reconstruction error, as well as the average number of iterations.

5.4.1 Noiseless Recovery of Low Rank Tensors

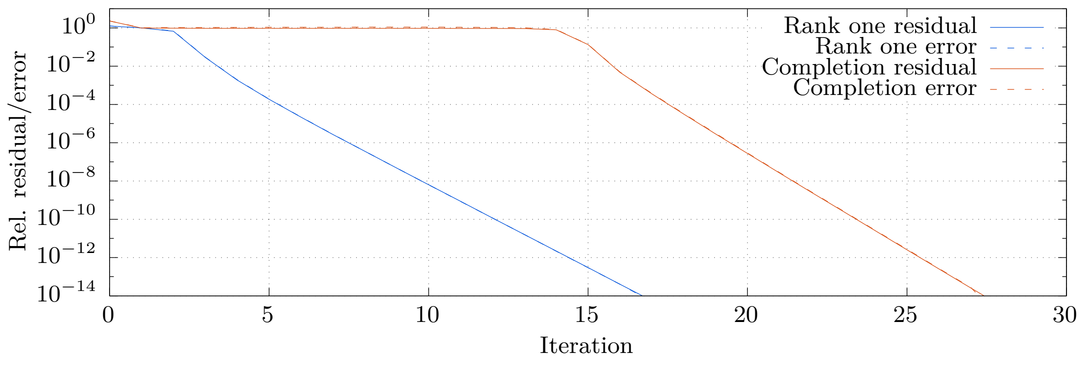


Figure 5.1: Performance of the alternating steepest descent algorithm for low rank tensor recovery of order $d = 7$ tensors with local dimensions $n = 10$ and rank $\boldsymbol{r} = (4, 2, 5, 3, 4, 2)$. The reconstruction uses $N = 100\,000$ rank one samples or entries respectively.

In this first experiment, we examine the recovery of random low rank tensors from noiseless measurements. Figure 5.1 shows the relative residual and the relative reconstruction error during the iteration of a single ASD run for the parameters $d = 7$, $n = 10$ and rank $\boldsymbol{r} = (4, 2, 5, 3, 4, 2)$ using $N = 100\,000$ rank one samples or entries, respectively. For both types of measurements, we observe that during the first few iterations there is no improvement in the residual or the reconstruction error. After this initial phase, we observe fast linear convergence down to machine precision, as expected for a (projected) gradient descent. As a result of the rather high number of samples, almost no difference between the residual and the reconstruction error is visible in this setting.

While there are limited theoretical results available, one of the most important questions is how many measurements are needed in practice for successful reconstruction. This is examined in the second setup, where we conduct several runs of the ASD for different numbers of samples. Figure 5.2 shows the ratio of successful reconstructions of order $d = 5$ tensors with local dimensions $n = 10$ for different ranks $\boldsymbol{r} = (r, \ldots, r)$. For rank one samples, we observe the presence of two distinct domains. The “few samples” domain, in which the reconstruction invariably fails and the “many samples” domain, where all reconstructions are successful. Between these two domains, there is a rather small transition area, where only some reconstructions are successful. The number of samples at which this transition happens, i.e. the number of samples required for successful reconstruction, is roughly related to the degrees of freedom the target tensor has according to its TT-rank (plotted in red), although larger by at least a factor of two. For the completion setting,

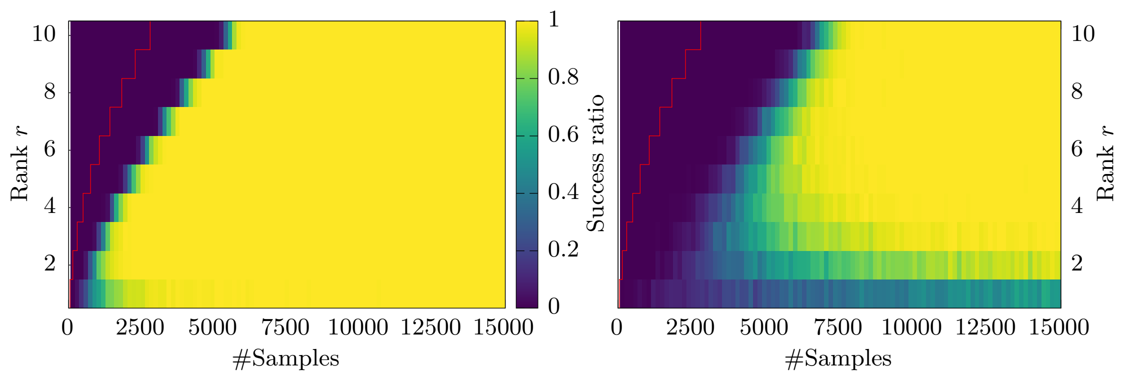


Figure 5.2: Ratio of successful noiseless reconstructions of order $d = 5$ tensors with local dimensions $n = 10$ for different ranks and number of samples. The red line denotes the degrees of freedom. **Left:** Rank one samples. **Right:** Completion setting.

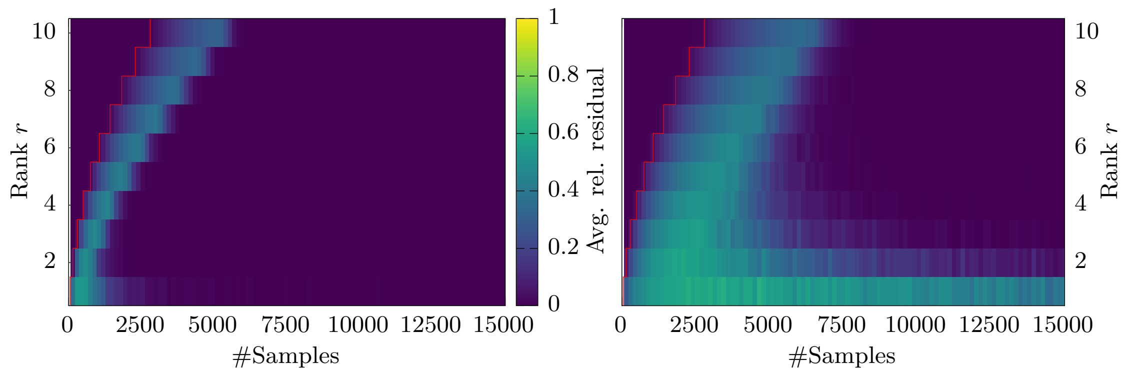


Figure 5.3: Avg. rel. residual norm of noiseless reconstructions of order $d = 5$ tensors with local dimensions $n = 10$ for different ranks and number of samples. The red line denotes the degrees of freedom. **Left:** Rank one samples. **Right:** Completion setting.

the two domains also exist, but the transition is less sharp, i.e. there is a significant region in which some reconstructions fail and some are successful. A major difference is that even for a large number of samples, the reconstruction for very small ranks often fails. In fact, for rank $r = 1$, a large portion of the reconstructions fail for all numbers of samples considered. This difficulty with rank $r = 1$ reconstructions is also visible for rank one samples, however much less distinct. For all ranks, successful reconstruction in the completion setting requires more samples than using rank one samples, but the amount is again roughly related to the degrees of freedom. The observation that very small ranks pose a problem in the completion setting is consistent with results of Grasedyck et al. [60], where a similar behavior can be observed for the ADF algorithm.

As established in Section 5.2, no unique reconstruction with fewer samples than the degrees of freedom is possible. However, in Figure 5.2, it is directly apparent that a successful reconstruction requires more than these degrees of freedom. One may ask whether this is due to the fact that the tensor is not uniquely reconstructable, or due to a shortcoming of the algorithm. A partial answer to this can be given by examining the value of the (normalized) target functional, i.e. the relative residual norm $J(\mathcal{X}) = \|\mathcal{T} - \mathcal{X}\|/\|\mathcal{T}\|$, which is shown in Figure 5.3. It is clearly visible that for fewer samples than required due to the degrees of freedom, the average residual is almost zero. This means that the algorithm finds a solution, which completely satisfies the measurements, but is different from the target tensor (otherwise the reconstructions were successful). In the domain, in which all reconstructions are successful, the avg. residual norm is of course zero as well, because the target tensor is found by the algorithm. The interesting part is in-between, where we in fact observe a rather large average residual norm. Note that in the noiseless setting, there always exists *a* global minimum, namely the target tensor $\mathcal{T}$, for which the residual vanishes. Every solution with non-zero residual is therefore at most a local minimum. This means that the ASD algorithm indeed gets stuck in local minima in this regime, however, we can only speculate whether the global minimum is unique.

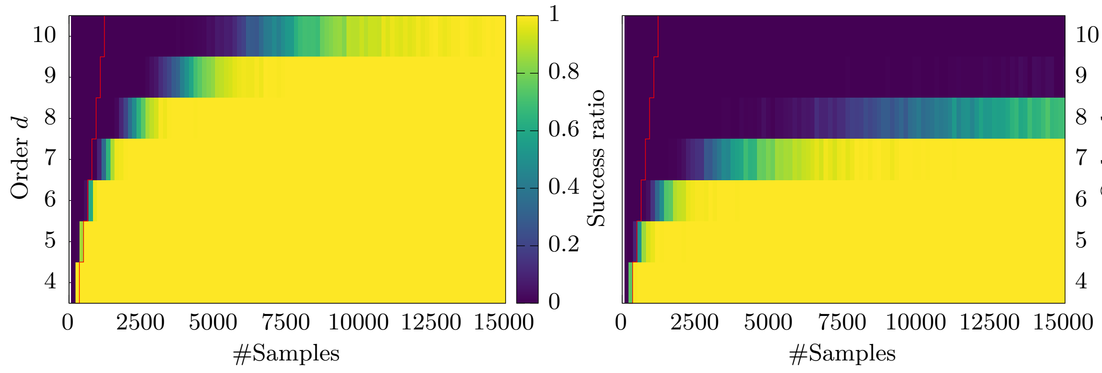


Figure 5.4: Ratio of successful noiseless reconstructions of rank $r = 4$ tensors with local dimensions $n = 10$ for different orders and number of samples. The red line denotes the degrees of freedom. **Left:** Rank one samples. **Right:** Completion setting.

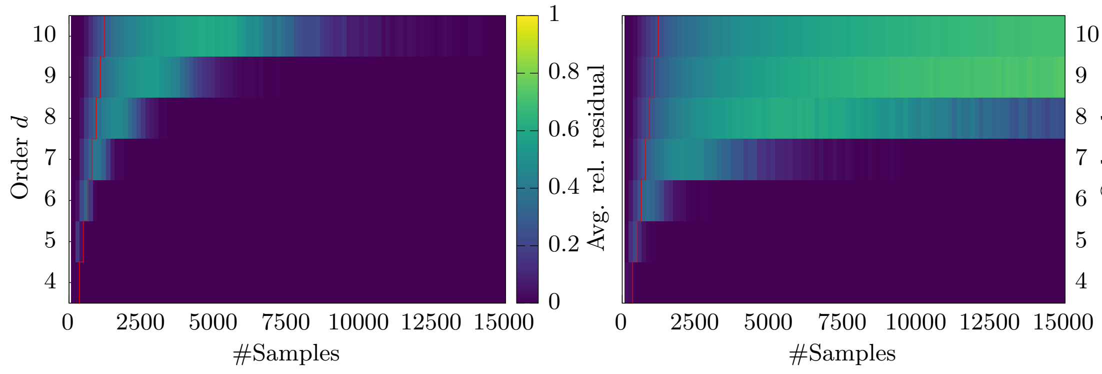


Figure 5.5: Avg. rel. residual norm of noiseless reconstructions of rank $r = 4$ tensors with local dimensions $n = 10$ for different orders and number of samples. The red line denotes the degrees of freedom. **Left:** Rank one samples. **Right:** Completion setting.

In the third setup, we examine the impact of the order on the reconstruction, i.e. instead of the rank, the order of the target tensor is varied. The local dimensions are chosen as $n = 5$ and the rank is $r = 4$. Figure 5.4 shows the ratio of successful reconstructions and Figure 5.5 shows the average relative residual norm of the solutions. The observations are largely analogous to the second setup, in particular, there are two distinct domains, where either all or no reconstitution is successful, with a transition area in-between. Again, the residual norm vanishes if there are fewer samples than degrees of freedom (barely visible) and if the reconstruction is successful. For the sample set sizes in-between, the residual norm is significantly non-zero, indicating that the algorithm gets stuck in local minima. The major observation is that the algorithm seems to struggle more for larger orders, as the number of measurements required for successful reconstruction increases faster with the order than expected from the degrees of freedom. This is especially striking for the completion setting, where there are no successful reconstructions for large orders within the considered range. However, note that for order 10, there are almost 10 million entries in the tensor, i.e. the range tops out at just 0.15% of measured entries.

5.4.2 Noisy recovery of low rank tensors

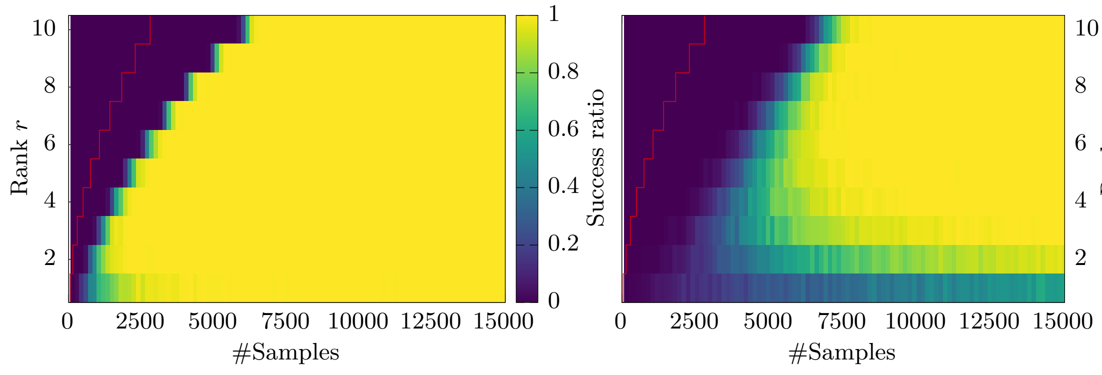


Figure 5.6: Ratio of successful noisy reconstructions of order $d = 5$ tensors with local dimensions $n = 10$ for different ranks and number of samples. The red line denotes the degrees of freedom. **Left:** Rank one samples. **Right:** Completion setting.

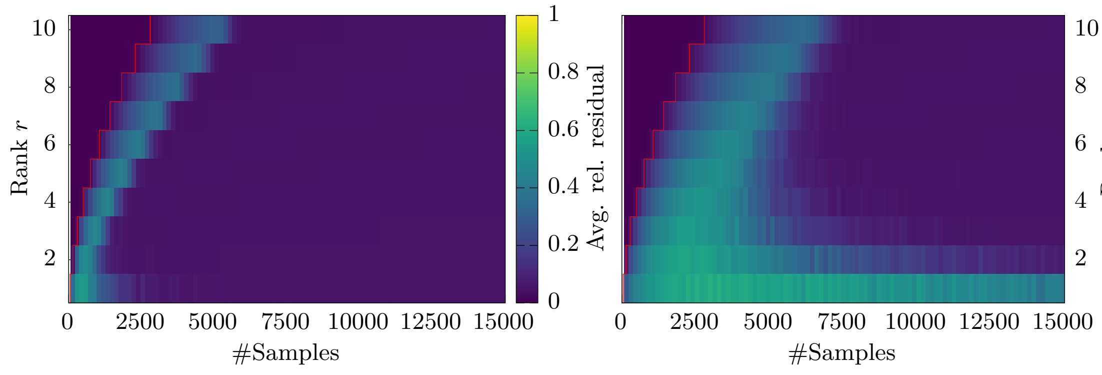


Figure 5.7: Avg. rel. residual norm of noisy reconstructions of order $d = 5$ tensors with local dimensions $n = 10$ for different ranks and number of samples. The red line denotes the degrees of freedom. **Left:** Rank one samples. **Right:** Completion setting.

In this second experiment, we examine the impact of noise in the measurements on the reconstruction quality. To this end, we conduct several runs of the ASD algorithm for random low rank target tensors and random measurements which are corrupted by noise with amplitude $\tau = 0.05$ (see (5.26) and (5.27)). The parameters are chosen as $d = 5$ and $n = 10$. Figure 5.6 shows the ratio of denoising reconstructions, i.e. those for which the relative reconstruction error is smaller than 0.05, for various ranks. Figure 5.7 shows the corresponding average relative residual norm of the solutions. The results are largely analogous to the noiseless setting, with the difference that slightly more samples are required for successful (denoising) reconstruction and that naturally, the residual norm of the solutions in the region of successful reconstruction is not zero, but somewhat below 0.05.[7] Overall, this shows that the reconstruction is stable with respect to noise and is in fact able to perform effective denoising.

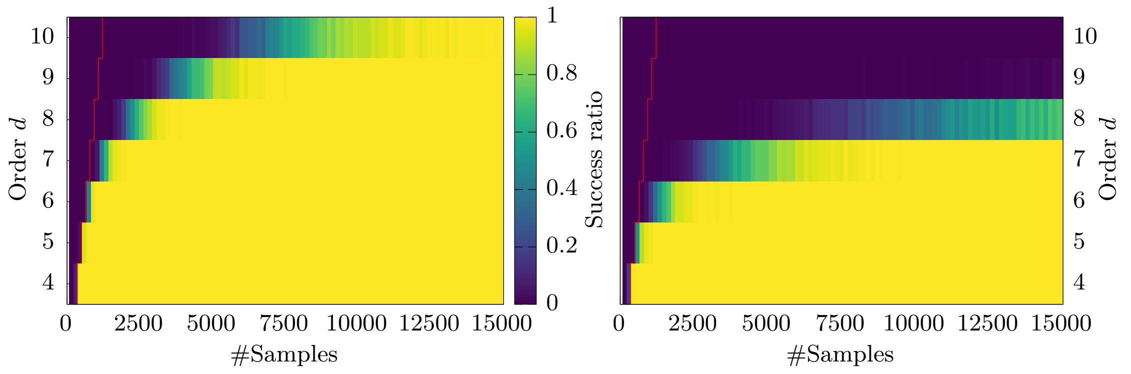


Figure 5.8: Ratio of successful noisy reconstructions of rank $r = 4$ tensors with local dimensions $n = 10$ for different orders and number of samples. The red line denotes the degrees of freedom. **Left:** Rank one samples. **Right:** Completion setting.

In the second setup, we again vary the order instead of the rank. Figure 5.8 shows the ratio of denoising reconstructions for $n = 5$ and $r = 4$. As for the first setup, there are only slight differences to the noiseless setting, indicating that the robustness to noise is independent of the order.

5.4.3 Recovery of tensors with decreasing singular values

In this next experiment, we use target tensors with approximately quadratically decaying singular values instead of exact low rank tensors. As the target tensors are not exactly of low rank, the notion of successful reconstructions is not clear in this setup. However, as the target tensors can be well approximated by low rank tensors, it makes sense to regard the relative reconstruction error. Figure 5.9 shows this for $d = 5$, $n = 10$ and various reconstruction ranks and samples sizes. As expected, the reconstruction error decreases with the reconstruction rank as the higher ranks allows better approximations of the target tensors. For rank one samples, similar to the previous experiments, there is a certain rank dependent threshold for the number of samples at which the reconstruction is successful, which in this case means that the average reconstruction error drops from almost one to

[7]Note that in this noisy setting, the target tensor $\mathcal{T}$ also incurs a relative residual of around 0.05

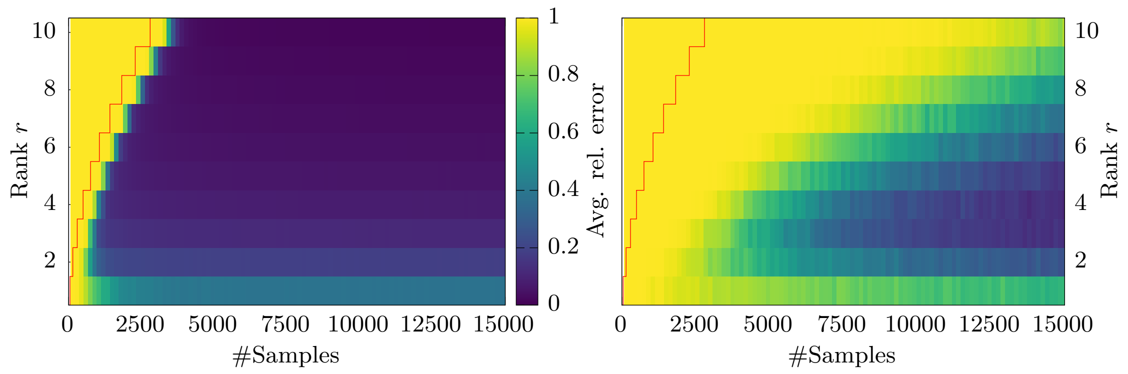


Figure 5.9: Avg. rel. reconstruction error for order $d = 5$ tensors with local dimensions $n = 10$ and approximately quadratically decreasing singular values for different ranks and number of samples. **Left:** Rank one samples. **Right:** Completion setting.

a value corresponding mostly to the rank. Further measurements beyond that threshold have only a minor effect. For the completion setting, there is almost no clear threshold, as the algorithm often fails to find good low rank approximations.

5.4.4 Recovery Using the Block ASD

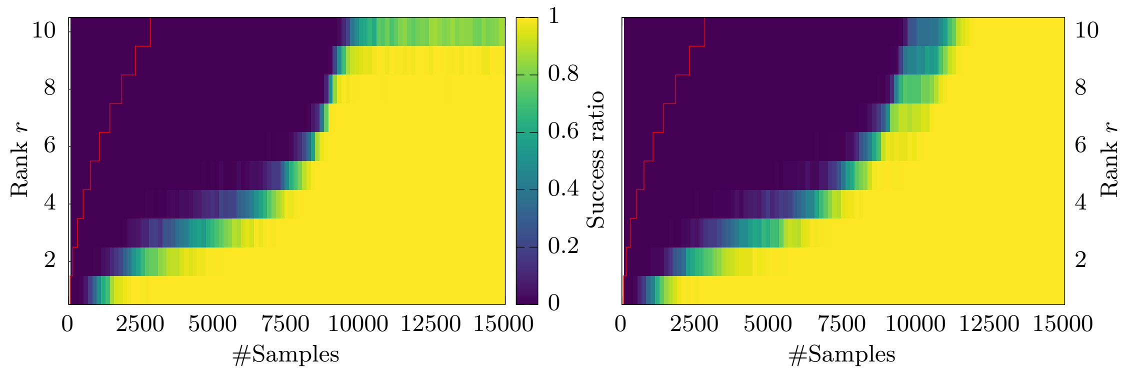


Figure 5.10: Ratio of successful reconstructions using the block-ASD of order $d = 5$ tensors with local dimensions $n = 10$ for different ranks and number of samples. The red line denotes the degrees of freedom. **Left:** Noiseless rank one samples. **Right:** Noisy rank one samples.

In this final experiment, we use the rank adaptive block ASD for the reconstruction. As in the other experiments, we run several reconstructions of random target tensors with random measurements for each parameter combination. Figure 5.10 shows the ratio of successful reconstructions for $d = 5$, $n = 10$ and various ranks for noiseless and noisy measurements using rank one samples. The noise amplitude is chosen as $\tau = 0.05$. Figure 5.11 shows the corresponding average relative residual norm of the solutions. Provided that sufficient samples are available, the reconstruction is successful for almost all ranks, however, the amount of samples required is significantly higher than for the fixed rank ASD used in the previous experiments. A notable exception is for rank 10 and noiseless

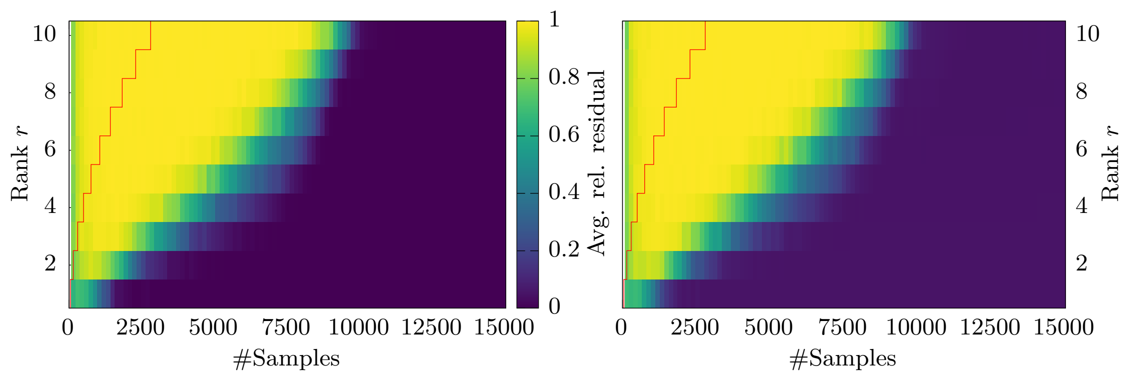


Figure 5.11: Avg. rel. residual norm of reconstructions using the block-ASD of order $d = 5$ tensors with local dimensions $n = 10$ for different ranks and number of samples. The red line denotes the degrees of freedom. **Left:** Noiseless rank one samples. **Right:** Noisy rank one samples.

measurements, where the success ratio remains somewhat below 0.9. The reason here is that sometimes, the algorithm finds a solution with slightly smaller rank,[8] which yields a relative residual norm of around 10^{-3} (also apparent from the vanishing average residual in Figure 5.11), i.e. less than the accuracy parameter ϵ. Consequently, the algorithm does not increase the rank and does not reach the threshold of 10^{-5} for successful reconstruction. Otherwise, the results are qualitatively similar to the fixed rank ASD, in particular the two distinct domains with a small transition area exist. An obvious difference, however, is that for fewer samples than required by the degrees of freedom, the block-ASD does not converge to a wrong but compatible solution, as this is directly detected and prevented by use of the control set.

5.5 Summary and Conclusions

In this chapter, we have recapitulated well-established results for low rank matrix recovery and used these as a basis to formulate different generalizations of the recovery problem from matrices to higher order tensors. We collected theoretical results, discussed advantages and disadvantages of the different approaches and provided an overview of available reconstruction algorithms. We derived the alternating steepest descent algorithm as a projected gradient procedure for tensor reconstruction using single entries or rank-one samples and introduced a novel rank adaptation technique for tensor recovery based on the block-TT format. Finally, we conducted extensive numerical experiments examining the performance of the proposed (block-) ASD algorithm for tensor recovery.

Thanks to recent theoretical results, e.g. by Rauhut et al. [61, 65] and Ashraphijuo and Wang [67], there are now some guarantees for the unique reconstructability of low rank tensors for certain classes of (random) measurements. In contrast to the matrix recovery problem, however, tensor recovery still lacks a tractable convex relaxation of the rank

[8]Note that we do *not* restrict the block-ASD to uniform TT-ranks

minimization problem, which preserves these recovery guarantees. A possible remedy are optimization algorithms on the low rank tensor manifold, combined with rank adaptation schemes where necessary. In this chapter, the alternating steepest descent algorithm was derived as a projected gradient descent as one such algorithm on the low rank tensor manifold. The numerical results show that this algorithm is indeed able to reconstruct different types of (approximately) low rank tensors from incomplete measurements and that it is robust with respect to noise. However, they also show that especially for high orders, the algorithm presumably gets stuck in local minima if an insufficient number of samples is provided. As the ASD is a local optimization algorithm, this is not unexpected, instead it is rather surprising that local minima are apparently no problem at all if a sufficient number of samples are provided. As a subject of future work, it would be interesting to explore whether the incorporation of global optimization techniques such as basin hopping and thereby the avoidance of local minima could reduce the number of required samples.

A novel approach taken in this work is the use of rank-one samples, which offer several benefits. First and foremost, rank-one samples represent a significant generalization of the completion setting, thereby enabling the use of more general types of measurements. This will for example be of use in the next chapter, where rank one samples appear naturally in an uncertainty quantification problem. In contrast to the even more general setting using an arbitrary $\mathcal{A} \in \mathbb{R}^{N \times n_1 \times \ldots \times n_d}$ to define the measurement operator, the use of rank one samples remains computationally feasible. In particular, the storage complexity as well as the computational complexity of an ASD iteration scales only linearly in the order of the target tensor, as shown in the previous sections. As a practical benefit, we observed in all numerical experiments that significantly fewer random rank one samples are required for successful reconstruction than entries in the completion setting. It remains a topic for future work to examine whether this observation can be backed by theoretical results on rank one samples, e.g. regarding the TRIP.

Another novelty presented in this work is the technique of splitting the measurements, allowing rank adaptation via the block-TT format. The results for this technique in the synthetic benchmark problems presented in this chapter show that the block-ASD is indeed able to derive the correct rank during the reconstruction without any a priori information. However, compared to the fixed rank ASD and other techniques, for example by Grasedyck and Krämer [66], this requires significantly more samples. In the author's estimation, the main obstacle is the adaptation of the accuracy ϵ used to control the rank increase and decrease. The results presented in this chapter all use a fixed control of ϵ, however, tests using an ϵ adapted to the parameters (d, n and r) show that much better results in terms of the required samples can be obtained. Therefore, using a more fine-grained and adaptive control of the parameter ϵ could possibly allow to significantly reduce the number of additional samples required compared to the fixed rank ASD and other techniques. This adaptive control is an important topic for future work. However, the block-ASD, as presented in this chapter, already proved to be a robust tool for the recovery problems appearing in the next chapter, where it will also be used. Let us finally note that this rank adaptation technique is not limited to the ASD algorithm, but can be used for most optimization algorithms for tensor recovery with rank one samples or single entries. Especially an application to the ALS algorithm could prove beneficial and will be explored in future work.

6 Application to Uncertainty Quantification and Variational Monte Carlo

In this chapter, we examine the application of tensor recovery techniques to uncertainty quantification (UQ) problems. To this end, Section 6.1 provides a brief introduction to uncertainty quantification and the parametric setting considered in this work. Motivated by this practical application, Section 6.2 gives an abstract formalism for variational Monte Carlo type reconstruction problems, which reveals some interesting connections between tensor recovery and machine learning. Section 6.3 is concerned with the practical realization of the reconstruction and presents a specialized version of the alternating steepest descent algorithm. In Section 6.4, this algorithm is used to conduct several numerical experiments, where tensor recovery techniques are used to obtain a functional approximation of the complete solution manifold of parametric partial differential equations. Finally, Section 6.5 summarizes the results and gives an outlook on future work.

Parts of this chapter have previously been published in [147], together with M. Eigel, J. Neumann and R. Schneider and will be part of an upcoming publication [148], together with M. Eigel, R. Schneider and P. Trunschke. The (quasi) Monte Carlo samples for the parametric PDE settings used in Section 6.4 of this work were provided by M. Eigel and P. Trunschke.

6.1 Uncertainty Quantification

The goal of Uncertainty Quantification is to incorporate incomplete knowledge and random effects common in many real-world settings. Typical applications come from engineering and the natural sciences, where the inclusion of uncertainties in modeling and simulation has become a standard requirement. Here the source of uncertainty can be both, due to an inherent randomness of the problem (aleatoric), e.g. common in engineering applications due to deviations in a production process (material perturbations) or random nature phenomena, such as wind forces acting on the structure, as well as due to a lack of knowledge (epistemic), e.g. common in fluid dynamics due to unknown makeup of the medium, see for example the surveys by Najm [149] and Wojtkiewicz et al. [150]. Both kinds of uncertainty can often be included in form of random variables (e.g. coefficients, forces and domains) in the mathematical modeling of the problem as a partial differential equation (PDE). A classical approach would then be to evaluate statistics of the solution or any other quantity of interest from random draws of the stochastic data, as it is done with Monte Carlo sampling and modern variants such as Multilevel Monte Carlo (MLMC) and quasi Monte Carlo (QMC), see for example the work by Giles [151], Cliffe et al. [152], and Kuo et al. [153]. As an alternative, functional approximations seek to determine a surrogate model in some discretization of the function space of the problem. The most common approaches are non-intrusive interpolation methods based on sparse grids (Stochastic Collocation) and intrusive Galerkin methods (Stochastic FEM), see for example the work of Babuška et al. [154], Nobile et al. [155], Babuška et al. [156], Giesselmann et al. [157], and Eigel et al. [158]. These methods have the fundamental advantage that they allow a complete parameterized solution, from which most quantities of interest, as well as the (deterministic) solution for any instance of the random data, can be obtained inexpensively. This, however, comes at the cost that the computation of the parameterized solution can be significantly more involved. To alleviate this obstacle, different model reduction techniques, e.g. reduced basis methods, have to be employed.

In this chapter, we aim to attain such a full functional approximation of the entire parametric solution using a non-intrusive tensor reconstruction approach. Using conforming finite elements (FE) in the physical space and orthogonal polynomials in the parameter space (so-called generalized polynomial chaos) allows a tensorized basis of the complete space, through which the functional approximation can be identified with a single high dimensional coefficient tensor. However, the exponential scaling in the order – which is equivalent to the stochastic dimensions plus usually one spacial dimension – would in most cases render the direct representation of this tensor unfeasible. Therefore, we use the TT-format as a low rank tensor representations to approximate the solution tensor. In earlier works, for example by Khoromskij and Schwab [159], Matthies et al. [160], Espig et al. [161], Khoromskij and Oseledets [162], Dolgov et al. [163], and Bachmayr et al. [164], it was shown that the operators of the corresponding parametric PDEs often exhibit a low rank structure. Based on this observation, different approaches are possible to compute the solution tensor. For example, Eigel et al. [165] used the low rank structure of the operator to formulate a fully adaptive intrusive algorithm, which provides a low TT-rank representation of the parametric Galerkin solution. Empirically, they showed the important result that in many cases, the solution tensor can also be well approximated in the low TT-rank format. One drawback of their approach is that it requires the construction of a low rank representation of the parametric operator and right hand side, as well as an efficient algorithm to solve the resulting system in a low rank fashion. This in particular requires adjustments for each problem setting and prevents the use of existing, possibly highly optimized, deterministic PDE solvers. The latter property is the reason why such methods are referred to as intrusive. As a remedy to these restrictions, we propose a completely non-intrusive reconstruction algorithm, which attempts to recover the complete solution tensor from a set of Monte Carlo (MC) measurements of the solution. These MC samples can be obtained without modifications from any existing PDE solver of choice. Additionally, since we do not require a specific choice of measurements, this method can also be applied to already available data, without the need for recomputations. The practical reconstruction in our approach is based on the tensor recovery techniques introduced in Chapter 5. We refer the interested reader to the work of Espig et al. [166], Dolgov et al. [167], and Dolgov and Scheichl [168] for related approaches.

6.1.1 Parametrized Uncertainty Quantification Setting

This section provides a brief review of the formal setting of partial differential equations with stochastic parameters, as commonly considered in uncertainty quantification. More detailed introductions can for example be found in the work of Schwab and Gittelson [169] and Lord et al. [170], as well as in the work of Eigel et al. [158, 165, 171] and Eigel and Merdon [172] for the adaptive Galerkin setting.

The general goal is to obtain a feasible representation of the parametric solution $u(\boldsymbol{x}, \boldsymbol{y}) \in V$ of an abstract problem

$$\mathcal{D}(u, \boldsymbol{y}) = 0 \ . \tag{6.1}$$

Here, $\mathcal{D}$ encodes a model in a physical domain $D \subset \mathbb{R}^\kappa$ with $\kappa = 1, 2, 3$, and $\boldsymbol{y} \in \mathbb{R}^M$ is a M-dimensional parameter vector encoding the random data. In the following, we assume that the components of $\boldsymbol{y}$ correspond to independent random variables with an associated product parameter domain Γ and joint product density γ. A prime example of this setting, which is often considered, is the stochastic diffusion equation

$$\begin{aligned} -\boldsymbol{\nabla}(a(\boldsymbol{x}, \omega)\boldsymbol{\nabla} u(\boldsymbol{x}, \omega)) &= f(\boldsymbol{x}, \omega) \qquad & \boldsymbol{x} \in D, \omega \in \Omega \\ u(\boldsymbol{x}, \omega) &= 0 & \boldsymbol{x} \in \partial D, \omega \in \Omega \ , \end{aligned} \tag{6.2}$$

where $a : D \times \Omega \to \mathbb{R}$ is a random field with finite variance depending on the probability space $(\Omega, \Sigma, \mathbb{P})$. Here, Ω is a sample space, Σ a σ-algebra on Ω and $\mathbb{P}$ a probability measure on Σ. The use of a classical truncated Karhunen-Loève expansion admits a representation of $a(\boldsymbol{x}, \omega)$ in terms of mutually uncorrelated random variables $y_\nu(\omega)$, see e.g. the work of Schwab and Gittelson [169], Loève [173], and Schwab and Todor [174]. In the numerical experiments in Section 6.4, we in particular examine the two prototypical cases of affine and lognormal dependence on $\boldsymbol{y}$, such that either

$$a(\boldsymbol{x}, \boldsymbol{y}) = a_0(\boldsymbol{x}) + \sum_{\nu=1}^{M} a_\nu(\boldsymbol{x}) y_\nu, \tag{6.3}$$

with $\Gamma = [-1, 1]^M$ and $y_\nu \sim \mathcal{U}(-1, 1)$, $\nu = 1, \dots, M$, or

$$a(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(\sum_{\nu=1}^{M} a_\nu(\boldsymbol{x}) y_\nu\right), \tag{6.4}$$

with $\Gamma = \mathbb{R}^M$ and $y_\nu \sim \mathcal{N}(0, 1)$, $\nu = 1, \dots, M$, where $\mathcal{U}(-1, 1)$ is the uniform distribution on $[-1, 1]$ and $\mathcal{N}(0, 1)$ is the normal distribution with zero mean and variance one. Using this representation of a allows a reformulation of the stochastic diffusion equation (6.2) as a parametric PDE with parameters $\boldsymbol{y}$, fitting to the aforementioned setting.

For most models, the solution $u \in V = X \otimes Y$, usually with $X = H_0^1(D)$ and $Y = L^2(\Gamma, \gamma)$, is infinite-dimensional and not directly accessible. This problem is solved by the use of finite-dimensional subspaces of X and Y. For X, the common choice is the use of a conforming finite element space $X_\rho(\tau) = \operatorname{span}\{\varphi_j\}_{j=1,\dots,S} \subset X$ of order ρ on a triangulation τ of the physical domain D. Since Γ exhibits a Cartesian product structure, the space Y is the tensor product of the Hilbert spaces $\{L^2(\Gamma_1, \gamma_1), \dots L^2(\Gamma_M, \gamma_M)\}$ endowed with a cross-norm, i.e. $Y = \bigotimes_{\nu=1}^M Y_\nu = \bigotimes_{\nu=1}^M L^2(\Gamma_\nu, \gamma_\nu)$. For each of the spaces $Y_\nu := L^2(\Gamma_\nu, \gamma_\nu)$, we use a truncated orthogonal polynomial basis, i.e. $\{P_\nu^{(q)}\}_{q=1,\dots,n_\nu}$, where $P_\nu^{(q)}$ is a univariate polynomial of degree $q-1$ and $\mathbb{E}_{\gamma_\nu}[P_\nu^{(q)} P_\nu^{(l)}] = \delta_{q,l}$ holds for all $1 \leq q, l \leq n_\nu$. For the normal and uniform distributions considered in the following, this means that $\{P_\nu^{(q)}\}_{q=1,\dots,n_\nu}$ is either a truncated Hermite or Legendre basis. Using $Y_\nu^{(n_\nu)} = \operatorname{span}\{P_\nu^{(q)}\}_{q=1,\dots,n_\nu} \subseteq L^2(\Gamma_\nu, \gamma_\nu)$ and Proposition 2.5, we obtain a finite-dimensional subspace

$$\tilde{V} = X_\rho(\tau) \otimes Y_{n_1}^{(1)} \otimes \dots \otimes Y_{n_M}^{(M)} \subseteq X \otimes Y \simeq V$$

of the original space. From Proposition 2.4, we know that there exists the tensorized basis

$$\tilde{V} = \operatorname{span}\Big\{\varphi_j \otimes P_1^{(q_1)} \otimes \dots \otimes P_M^{(q_M)} \;\Big|\; j = 1, \dots, S;\; q_\nu = 1, \dots, n_\nu\Big\} \,.$$

An approximation $\tilde{u}$ in the finite-dimensional subspace $\tilde{V} \subset V$ can then be given in this basis as

$$u(\boldsymbol{x}, \boldsymbol{y}) \approx \tilde{u}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{S} \sum_{q_1}^{n_1} \dots \sum_{q_M}^{n_M} \mathcal{T}[j, q_1, \dots, q_M]\; \varphi_j(\boldsymbol{x})\; P_1^{(q_1)}(y_1) \dots P_M^{(q_M)}(y_M) \;, \tag{6.5}$$

for a coefficient tensor $\mathcal{T} \in \mathbb{R}^{S \times n_1 \times \dots \times n_M}$ of order $M+1$. Note that for fixed bases, $\tilde{u}(\boldsymbol{x}, \boldsymbol{y})$ is completely defined by the coefficient tensor, which is the object that we aim to reconstruct in the following sections. The determination of suitable truncation thresholds n_ν and the adaptation of the FE space in the above setting is discussed by Eigel et al. [165]. Defining the vectors

$$\boldsymbol{\Phi}(\boldsymbol{x}) := \begin{pmatrix} \varphi_1(\boldsymbol{x}) \\ \varphi_2(\boldsymbol{x}) \\ \vdots \\ \varphi_S(\boldsymbol{x}) \end{pmatrix} \in \mathbb{R}^S \quad \text{and} \quad \boldsymbol{\xi}_\nu(y) := \begin{pmatrix} P_\nu^{(1)}(y) \\ P_\nu^{(2)}(y) \\ \vdots \\ P_\nu^{(n_\nu)}(y) \end{pmatrix} \in \mathbb{R}^{n_\nu} \;,$$

equation (6.5) can be phrased as

$$\tilde{u}(\boldsymbol{x}, \boldsymbol{y}) = \mathcal{T} \circ_{M+1} (\boldsymbol{\Phi}(\boldsymbol{x}) \otimes \boldsymbol{\xi}_1(y_1) \otimes \dots \otimes \boldsymbol{\xi}_M(y_M)) \;. \tag{6.6}$$

6.2 Tensor Recovery Approach and the Variational Monte Carlo Framework

This section introduces the idea of our tensor recovery approach for the uncertainty quantification setting. Before turning towards the practical aspects, we examine this UQ recovery problem from an abstract perspective and explore relations to machine learning techniques. This abstract variational Monte Carlo formulation is more general than the UQ problem at hand and may also be applied to a variety of other applications. This will also be covered in detail in a separate future publication [148] together with M. Eigel, R. Schneider and P. Trunschke.

The goal of the recovery approach is to find a function

$$\tilde{v}(\boldsymbol{x}, \boldsymbol{y}) = \mathcal{X} \circ_{M+1} (\boldsymbol{\Phi}(\boldsymbol{x}) \otimes \boldsymbol{\xi}_1(y_1) \otimes \ldots \otimes \boldsymbol{\xi}_M(y_M)) \in \tilde{V} \ ,$$

which approximates the sought-for solution $\tilde{u}(\boldsymbol{,y})$. Using $n := \max(n_\nu)$, the dimension of $\tilde{V}$ scales as $\Theta(Sn^M)$, i.e. exponentially in the number of parameters. Therefore, a direct approximation in $\tilde{V}$ is usually unfeasible, as it would require at least that many measurements for a unique recovery and accordingly, an exponential amount of storage for the solution. Instead, we restrict the approximate solution to the set

$$\tilde{\mathcal{M}}_{\preceq \boldsymbol{r}} := \left\{ \mathcal{Y} \circ_{M+1} (\boldsymbol{\Phi}(\boldsymbol{x}) \otimes \boldsymbol{\xi}_1(y_1) \otimes \ldots \otimes \boldsymbol{\xi}_M(y_M)) \;\middle|\; \mathcal{Y} \in \mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}} \right\}$$

of functions from $\tilde{V}$ for which the coefficient tensor in the tensorized basis has TT-rank at most $\boldsymbol{r}$. The dimension of $\tilde{\mathcal{M}}_{\preceq \boldsymbol{r}}$ then scales as the dimension of the corresponding TT-manifold, i.e. $\mathcal{O}(Sr + Mnr^2)$. As we want $\tilde{v}$ to approximate $\tilde{u}$ for *all* possible parameters $\boldsymbol{y}$, a reasonable measure of the quality of the approximation is the cost functional

$$J(\tilde{v}) := \int_\Gamma \|\tilde{v}(\cdot, \boldsymbol{y}) - \tilde{u}(\cdot, \boldsymbol{y})\|_c^2 \,\mathrm{d}\gamma(\boldsymbol{y}) \tag{6.7}$$

for some X-norm $\|\cdot\|_c$. However, unless we already have a data-sparse representation of $\tilde{u}$ at hand, evaluating this cost functional is usually unfeasible, as it requires knowledge of the solution for all parameter combinations. Instead, we aim to only use samples of the solution for some fixed parameters $\boldsymbol{y}^{(k)}$. That is, we assume that, for a set $\{\boldsymbol{y}^{(1)}, \ldots \boldsymbol{y}^{(N)}\}$ of N parameter vectors, the corresponding solutions $\tilde{u}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$ are known. In order to avoid the evaluation of the complete solution in the cost functional (6.7), we can then use the *empirical cost functional*

$$J_N(\tilde{v}) := \frac{1}{N} \sum_{k=1}^{N} \left\| \tilde{v}\Big(\cdot, \boldsymbol{y}^{(k)}\Big) - \tilde{u}\Big(\cdot, \boldsymbol{y}^{(k)}\Big) \right\|_c^2$$

to find an approximation of $\tilde{u}$. As this functional only requires knowledge of the solution $\tilde{u}$ for the known parameters $\boldsymbol{y}^{(k)}$, we can indeed optimize with respect to this functional using the information at hand. Note that for fixed $\boldsymbol{y}^{(k)}$, the problem (6.1) is deterministic and not parameter-dependent. Therefore, contrary to the complete parametric solution, the solutions $\tilde{u}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$ are usually inexpensive to obtain. For example, if $\mathcal{D}$ encodes the stochastic diffusion equation, then $\tilde{u}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$ is the solution to the deterministic diffusion equation using the deterministic field $a(\boldsymbol{x}, \boldsymbol{y}^{(k)})$, which can be obtained by any PDE solver of choice.

6.2.1 Variational Monte Carlo

The above setting can be embedded in a more general variational Monte Carlo framework, which allows a systematic treatment of the occurring errors. For this, let H be a Hilbert space and let (Ω, Σ, ρ) be a probability space. We consider variational problems defined by a *cost functional*

$$J(\Psi) := \int_\Omega \ell(\Psi, z) \,\mathrm{d}\rho(z) = \mathbb{E}[\ell(\Psi, \cdot)] \ , \tag{6.8}$$

which can be cast into the form of an integral over a *loss function* $\ell : H \times \Omega \to \mathbb{R}$. We assume that $\ell(\Psi, \cdot)$ is integrable with respect to the measure ρ for every $\Psi \in H$. The goal is to find a minimizer

$$\Psi^* \in \operatorname*{argmin}_{\Psi \in H} J(\Psi) \ .$$

As for the practical UQ application above, it is usually essential to restrict the minimization to a compact[1] *model class* $\mathcal{M} \subseteq H$ and consider the constrained minimization problem

$$\Psi^*_{\mathcal{M}} \in \operatorname*{argmin}_{\Psi \in \mathcal{M}} J(\Psi) \ .$$

On the one hand, this serves the purpose of making the numerical computations feasible, as in many setting the space H is very large or even infinite-dimensional. On the other hand, the restriction to the model class $\mathcal{M}$ allows reconstructions using only a limited number of samples, as we will outline in the following.

Computing the exact integral in (6.8) is often unfeasible even for a given Ψ, let alone performing an optimization on this basis. However, it is reasonable to assume that independent samples $\{z^{(k)}\}_{1 \le k \le N}$, distributed according to ρ, can be generated. For this setting, we propose to resort to Monte Carlo integration for the calculation of the expected value and use the *empirical cost functional*

$$J_N(\Psi) := \frac{1}{N} \sum_{k=1}^{N} \ell\Big(\Psi; z^{(k)}\Big) = \mathbb{E}[\ell(\Psi, \cdot); N] \ ,$$

which provides a surrogate functional that can be minimized to obtain

$$\Psi^*_{(\mathcal{M},N)} \in \operatorname*{argmin}_{\Psi \in \mathcal{M}} J_N(\Psi) \ . \tag{6.9}$$

[1]Note that for the UQ application introduced at the beginning of the section, the model class $\tilde{\mathcal{M}}_{\preceq \boldsymbol{r}}$ is not compact itself. However, this can easily be accomplished by using the intersection $\mathcal{M}^{\mathrm{TT}}_{\preceq \boldsymbol{r}} \cap \mathcal{B}(0, \lambda)$ with a ball of sufficiently large radius λ in the definition of $\tilde{\mathcal{M}}_{\preceq \boldsymbol{r}}$.

This is now a typical machine learning problem in the spirit of Cucker and Smale [175]. Therefore, it is possible and beneficial to employ ideas from statistical learning theory for the analysis of the resulting approximation $\Psi^*_{(\mathcal{M},N)}$. In this statistical learning framework, it is expedient to split the error introduced by the use of the model class and the empirical cost functional into the following components:

$$\begin{aligned}\mathcal{E} :=& \left|J(\Psi^*) - J\left(\Psi^*_{(\mathcal{M},N)}\right)\right| \\ \leq& \underbrace{\left|J(\Psi^*) - J(\Psi^*_{M})\right|}_{:=\mathcal{E}_{\text{App}}} + \underbrace{\left|J(\Psi^*_{\mathcal{M}}) - J\left(\Psi^*_{(\mathcal{M},N)}\right)\right|}_{:=\mathcal{E}_{\text{Gen}}} + \underbrace{\left|J\left(\Psi^*_{(\mathcal{M},N)}\right) - J(\Psi_{\text{Opt}})\right|}_{:=\mathcal{E}_{\text{Opt}}} \ .\end{aligned}$$

Here, Ψ_{Opt} denotes the solution obtained by an optimization algorithm applied to solve (6.9). The interpretation of these components is as follows.

- $\mathcal{E}_{\text{App}}$ is the *approximation error*, which is the error incurred by the use of the model class $\mathcal{M}$, i.e. the error of the best approximation within the model class, in terms of the cost functional. This error naturally decreases with the size of the model class.
- $\mathcal{E}_{\text{Gen}}$ is the *generalization error*, which is the error caused by the use of the empirical cost function J_N instead of J. This error generally decreases with the number of samples N, as the Monte Carlo integration used in the definition of J_N becomes more and more accurate.
- $\mathcal{E}_{\text{Opt}}$ is the *optimization error*, which is the error due to the practical optimization algorithm failing to find an exact minimizer (6.9) of the empirical cost functional. This can be due to the accuracy of the used optimization algorithm and, usually more importantly, due to local minima, in which the optimization gets stuck. This error is highly dependent on the used algorithms and can rarely be estimated in general.

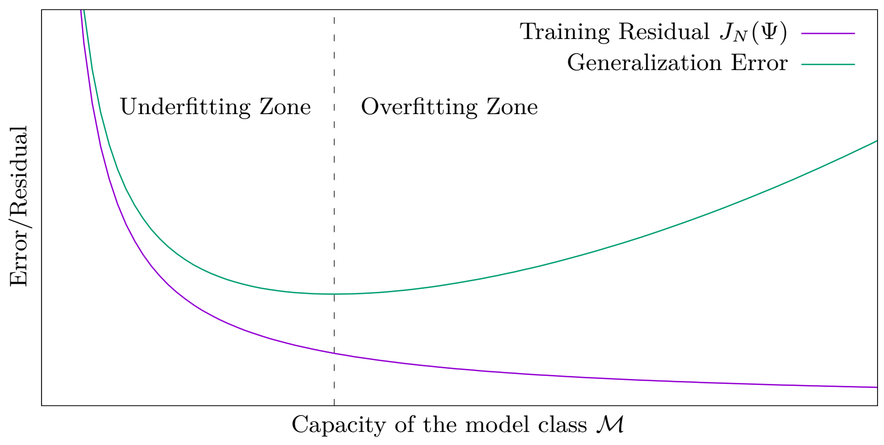


Figure 6.1: Typical relation of the training residual and the generalization error in dependence on the capacity of the model class $\mathcal{M}$ (see e.g. [176]). While the training residual generally decreases with the capacity, the generalization error is usually optimal for a limited capacity of the model class and worsens for larger sets due to overfitting.

An important aspect is the antagonistic relation between the approximation error and the generalization error. While enlarging the model class $\mathcal{M}$ naturally decreases the approximation error and the residual on the training data, it in turn generally increases the generalization error. This typical relation is depicted in Figure 6.1. Using tools from statistical learning theory, it is possible to give probabilistic bounds on these errors. Deriving such error bounds for the variational Monte Carlo framework of this section will be the topic of an upcoming publication [148], together with M. Eigel, R. Schneider and P. Trunschke. An important example of a result adapted from the work of Macdonald [177] is that the generalization error can be bounded using covering numbers defined as follows.

Definition 6.1 (Covering Number)**.** The covering number $\kappa(\mathcal{M},\, \epsilon)$ of a subset $\mathcal{M}$ of H is defined as the minimal number of $\|\cdot\|_H$-open balls of radius ϵ needed to cover $\mathcal{M}$.

For the set of low rank tensors used in the practical UQ application, Rauhut et al. [65] have shown that the covering numbers can be bounded by

$$\kappa\Big(\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\ldots\times n_d}) \cap \mathcal{B}(0,1),\, \epsilon\Big) \leq \left(\frac{3(2d-1)\sqrt{r}}{\epsilon}\right)^{\dim\left(\mathcal{M}_{\boldsymbol{r}}^{\mathrm{TT}}(\mathbb{R}^{n_1\times\ldots\times n_d})\right)} ,$$

where $r := \max_\mu(r_\mu)$. The result on the generalization error can then be stated as follows.

Theorem 6.2 (Generalization Error Bound [148])**.** *Assume that ℓ is bounded, i.e. there exists $C_1 > 0$ s.t. for all $\Psi \in \mathcal{M}$, it holds that*

$$|\ell(\Psi, z)| \leq C_1 \quad \textit{for almost all } z \in \Omega$$

and that ℓ is Lipschitz continuous on $\mathcal{M}$, i.e. there exists $C_2 > 0$ s.t. for all $\Psi_1, \Psi_2 \in \mathcal{M}$ it holds that

$$|\ell(\Phi_1, z) - \ell(\Psi_2, z)| \leq C_2\|\Psi_1 - \Psi_2\|_H \quad \textit{for almost all } z \in \Omega \quad .$$

Then, the following probabilistic error bound on the generalization error holds for all $\epsilon > 0$:

$$\mathbb{P}[\mathcal{E}_{gen} > \epsilon] \leq 2\kappa\Big(\mathcal{M}, \frac{\epsilon}{8C_2}\Big)e^{-\epsilon^2 N/(32C_1^2)} \ .$$

6.3 Reconstruction Algorithm

In this section, we show that for the UQ setting the variational Monte Carlo approach of the previous section can be cast as a tensor recovery problem. This allows the use of the tools from Chapter 5 and in particular the alternating steepest descent (ASD) algorithm to obtain the parametric approximation $\tilde{v}$ from the known solutions $\tilde{u}(\cdot, \boldsymbol{y}^{(k)})$. Furthermore, we show that a version of the ASD algorithm specialized to this UQ reconstruction problem exhibits a significantly superior scaling compared to a direct application of the ASD from Section 5.3.4, which allows a very efficient calculation of the parametric UQ solution.

In principle, alternating steepest descent algorithms can be used for different loss functions ℓ, however in the following we concentrate on the least squares loss functional $\ell(\tilde{v}, \boldsymbol{y}) = \|\tilde{v}(\cdot, \boldsymbol{y}) - \tilde{u}(\cdot, \boldsymbol{y})\|_c^2$, for some norm $\|\cdot\|_c$ on X. The empirical cost function is then given by

$$J_N(\tilde{v}) := \frac{1}{N} \sum_{k=1}^{N} \ell\Big(\tilde{v}, \boldsymbol{y}^{(k)}\Big) = \frac{1}{N} \sum_{k=1}^{N} \Big\| \tilde{v}\Big(\cdot, \boldsymbol{y}^{(k)}\Big) - \tilde{u}\Big(\cdot, \boldsymbol{y}^{(k)}\Big) \Big\|_c^2 . \tag{6.10}$$

Using the basis expansion from (6.6) for $\tilde{u}$ and analogously

$$\tilde{v}(\boldsymbol{x}, \boldsymbol{y}) = \mathcal{X} \circ_{M+1} (\boldsymbol{\Phi}(\boldsymbol{x}) \otimes \boldsymbol{\xi}_1(y_1) \otimes \ldots \otimes \boldsymbol{\xi}_M(y_M))$$

for $\tilde{v}$, (6.10) can be formulated in terms of the tensors $\mathcal{T}, \mathcal{X} \in \mathbb{R}^{S \times n_1 \times \ldots \times n_M}$ as

$$J_N(\tilde{v}) = \frac{1}{N} \sum_{k=1}^{N} \Big\| (\mathcal{X} - \mathcal{T}) \circ_{M+1} \Big(\boldsymbol{\Phi}(\cdot) \otimes \boldsymbol{\xi}_1(y_1^{(k)}) \otimes \ldots \otimes \boldsymbol{\xi}_M(y_M^{(k)}) \Big) \Big\|_c^2 .$$

Using the 2-norm defined by the values on the FE nodes for $\|\cdot\|_c$,[2] this simplifies to

$$J_N(\tilde{v}) = \frac{1}{N} \sum_{j=1}^{S} \sum_{k=1}^{N} \Big((\mathcal{X} - \mathcal{T}) \circ_{M+1} \Big(\boldsymbol{e}_j \otimes \boldsymbol{\xi}_1(y_1^{(k)}) \otimes \ldots \otimes \boldsymbol{\xi}_M(y_M^{(k)}) \Big) \Big)^2 .$$

Defining the rank-one measurement operator

$$\hat{\mathcal{A}} : \mathbb{R}^{S \times n_1 \times \ldots \times n_M} \to \mathbb{R}^{S \cdot N}$$
$$\hat{\mathcal{A}}(\mathcal{X})_{j,k} = \mathcal{X} \circ_{M+1} \left(\frac{1}{N} \boldsymbol{e}_j \otimes \boldsymbol{\xi}_1(y_1^{(k)}) \otimes \ldots \otimes \boldsymbol{\xi}_M(y_M^{(k)}) \right)$$

and the measured values as

$$\boldsymbol{b} := \hat{\mathcal{A}}(\mathcal{T}) \in \mathbb{R}^{S \cdot N} ,$$

we finally obtain

$$J_N(\tilde{v}) = \Big\| \hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b} \Big\|_2^2 .$$

This is exactly the tensor recovery setting with a rank-one measurement operator $\hat{\mathcal{A}}$. In principle, this formulation allows the direct application of the ASD algorithm as introduced in Section 5.3.4. However, note that this formulation treats each parameter combination of spatial and parametric position as individual measurements, hence there are NS rank one samples. This and the fact that the first mode of the solution has dimension S results in a computational complexity of $\mathcal{O}(Mnr^2NS + r^2NS^2)$ for one sweep of the unmodified ASD from Section 5.3.4, where $n = \max_\nu(n_\nu)$ and $r = \max_\nu(r_\nu)$. In the following, we show that an implementation of the ASD specialized to this setting is able to reduce the complexity to $\mathcal{O}(Mnr^2N + Mr^3N + rNS)$.[3] Especially for large S, i.e. a fine FE triangulation, this is a tremendous improvement.

Similar to the ASD implementation in Section 5.3.4, the foundation of the efficient implementation are stacks, which store partial contractions between the current iterate,

[2]Note that using this norm provides a particularly simple formulation, however, using for example the L^2 norm on the space X also works by including the corresponding stiffness matrix.

[3]In both statements, we assumed $N > r$ and neglected the Mnr^3 and r^2S terms due to the core movement.

the measurement operator and the tangent space projectors. Compared to the stacks defined in (5.21) and (5.22), the stacks defined here are more involved, as they have to absorb the possibly large spatial dimension S. In the following, we use the notation from Section 5.3.4, i.e. among other things we assume that the current iterate is given in an orthogonalized TT-representation

$$\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d \in \mathbb{R}^{S \times n_1 \times \ldots \times n_M} .$$

The core position is assumed to change, corresponding to the projector $\hat{P}_\mu$ under consideration. To simplify the notation, we combine all measurements with the same stochastic parameters and define the vectors $\tilde{\boldsymbol{b}}_k \in \mathbb{R}^S$ as $\tilde{\boldsymbol{b}}_k[j] := \boldsymbol{b}_{j,k}$ for $j = 1, \ldots, S$ and $k = 1, \ldots, N$. Under this premise, the stacks are defined as

$$\begin{aligned}
\mathcal{S}^{\text{LeftIs}}_{1,k} &:= \boldsymbol{I} && \in \mathbb{R}^{r_1 \times r_1} \quad (6.11)\\
\mathcal{S}^{\text{LeftIs}}_{\mu+1,k} &:= \Big(\boldsymbol{\xi}_\mu\Big(y^{(k)}_\mu\Big) *_{1,2} \mathcal{U}_{\mu+1}\Big) *_{1,1} \mathcal{S}^{\text{LeftIs}}_{\mu,k} *_{2,1} \Big(\boldsymbol{\xi}_\mu\Big(y^{(k)}_\mu\Big) *_{1,2} \mathcal{U}_{\mu+1}\Big) && \in \mathbb{R}^{r_{\mu+1} \times r_{\mu+1}}\\
\mathcal{S}^{\text{LeftOught}}_{1,k} &:= \tilde{\boldsymbol{b}}_k \circ \mathcal{U}_1 && \in \mathbb{R}^{r_1} \quad (6.12)\\
\mathcal{S}^{\text{LeftOught}}_{\mu+1,k} &:= \mathcal{S}^{\text{LeftOught}}_{\mu,k} \circ \Big(\boldsymbol{\xi}_\mu\Big(y^{(k)}_\mu\Big) *_{1,2} \mathcal{U}_{\mu+1}\Big) && \in \mathbb{R}^{r_{\mu+1}}\\
\mathcal{S}^{\text{Right}}_{M+1,k} &:= \boldsymbol{\xi}_M\Big(y^{(k)}_M\Big) *_{1,2} \mathcal{U}_{M+1} && \in \mathbb{R}^{r_M} \quad (6.13)\\
\mathcal{S}^{\text{Right}}_{\mu-1,k} &:= \Big(\boldsymbol{\xi}_{\mu-2}\Big(y^{(k)}_{\mu-2}\Big) *_{1,2} \mathcal{U}_{\mu-1}\Big) \circ \mathcal{S}^{\text{Right}}_{\mu,k} && \in \mathbb{R}^{r_{\mu-2}} ,
\end{aligned}$$

where the core position is assumed to be larger than ν in the definition of the left stacks $\mathcal{S}^{\text{LeftIs}}_{\nu,k}$ and $\mathcal{S}^{\text{LeftOught}}_{\nu,k}$ and smaller than ν in the definition of the right stacks $\mathcal{S}^{\text{Right}}_{\nu,k}$. Note that while artificial at first glance, this core position is naturally enforced in the resulting algorithm. The computational complexity for $\mathcal{S}^{\text{LeftIs}}_{\mu+1,k}$ is $\mathcal{O}(nr^2 + r^3)$, assuming that all components are given. For $\mathcal{S}^{\text{LeftOught}}_{1,k}$, it is $\mathcal{O}(rS)$ and for the other stacks, it is $\mathcal{O}(nr^2)$, again assuming that all components are given. Calculating all left stacks for $\mu = 1, \ldots, M$ and all right stacks for $\mu = M+1, \ldots, 2$ therefore has a total complexity bounded by $\mathcal{O}(Mnr^2N + Mr^3N + rNS)$.

The stacks can be used to inexpensively calculate the crucial components

$$\mathcal{W}_\mu := \mathcal{U}_{<\mu} *_{(1,\ldots,\mu-1),(1,\ldots,\mu-1)} \Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big) *_{(\mu+1,\ldots,d),(2,\ldots,d-\mu+1)} \mathcal{U}_{>\mu}$$

and $\Big\|\hat{\mathcal{A}}\Big(\hat{P}_\mu\Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big)\Big)\Big\|_2^2$ of the algorithm. In particular, one can straightforwardly verify that

$$
\begin{aligned}
\mathcal{W}_1 &= \underbrace{\sum_{k=1}^{N} \left(\mathcal{U}_1 \circ \mathcal{S}^{\text{Right}}_{2,k}\right) \otimes \mathcal{S}^{\text{Right}}_{2,k}}_{=\left(\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X})\right)\right) *_{(2,\ldots,d),(2,\ldots,d)} \mathcal{U}_{>1}} \quad - \quad \underbrace{\sum_{k=1}^{N} \tilde{\boldsymbol{b}}_k \otimes \mathcal{S}^{\text{Right}}_{2,k}}_{=\left(\hat{\mathcal{A}}^T(\boldsymbol{b})\right) *_{(2,\ldots,d),(2,\ldots,d)} \mathcal{U}_{>1}} \\
\mathcal{W}_\mu &= \underbrace{\sum_{k=1}^{N} \left(\mathcal{S}^{\text{LeftIs}}_{\mu-1,k} \circ \left(\boldsymbol{\xi}_{\mu-1}\left(y^{(k)}_{\mu-1}\right) *_{1,2} \mathcal{U}_\mu\right) \circ \mathcal{S}^{\text{Right}}_{\mu+1,k}\right) \otimes \boldsymbol{\xi}_{\mu-1}\left(y^{(k)}_{\mu-1}\right) \otimes \mathcal{S}^{\text{Right}}_{\mu+1,k}}_{=\mathcal{U}_{<\mu} *_{(1,\ldots,\mu-1),(1,\ldots,\mu-1)} \left(\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X})\right)\right) *_{(\mu+1,\ldots,d),(2,\ldots,d-\mu+1)} \mathcal{U}_{>\mu}} \qquad (6.14) \\
&\quad - \underbrace{\sum_{k=1}^{N} \mathcal{S}^{\text{LeftOught}}_{\mu,k} \otimes \boldsymbol{\xi}_{\mu-1}\left(y^{(k)}_{\mu-1}\right) \otimes \mathcal{S}^{\text{Right}}_{\mu+1,k}}_{=\mathcal{U}_{<\mu} *_{(1,\ldots,\mu-1),(1,\ldots,\mu-1)} \left(\hat{\mathcal{A}}^T(\boldsymbol{b})\right) *_{(\mu+1,\ldots,d),(2,\ldots,d-\mu+1)} \mathcal{U}_{>\mu}}
\end{aligned}
$$

and

$$
\begin{aligned}
&\left\|\hat{\mathcal{A}}\left(\hat{P}_1\left(\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\right)\right)\right)\right\|_2^2 = \sum_{k=1}^{N} \left\|\mathcal{W}_1 \circ \mathcal{S}^{\text{Right}}_{\mu+1,k}\right\|_F^2 \\
&\left\|\hat{\mathcal{A}}\left(\hat{P}_\mu\left(\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\right)\right)\right)\right\|_2^2 \qquad (6.15) \\
&= \sum_{k=1}^{N} \left(\left(\boldsymbol{\xi}_{\mu-1}\left(y^{(k)}_{\mu-1}\right) *_{1,2} \mathcal{W}_\mu\right) \circ \mathcal{S}^{\text{Right}}_{\mu+1,k}\right) \circ \mathcal{S}^{\text{LeftIs}}_{\mu-1,k} \circ \left(\left(\boldsymbol{\xi}_{\mu-1}\left(y^{(k)}_{\mu-1}\right) *_{1,2} \mathcal{W}_\mu\right) \circ \mathcal{S}^{\text{Right}}_{\mu+1,k}\right)
\end{aligned}
$$

hold, with the obvious exception that for $\mu = M + 1$, the right stack does not appear in either equation. The computational complexity is bounded by $\mathcal{O}(rS)$ for $\mathcal{W}_1$ and by $\mathcal{O}(nr^2)$ for the others, assuming that all components are given. The complete procedure constructing and using these stacks for an alternating steepest descent is given in Algorithm 11. The complete computational complexity for one sweep is bounded by $\mathcal{O}(Mnr^2N + Mr^3N + Mnr^3 + r^2S + rNS)$. As $N > r$ holds in all reasonable settings, the terms Mnr^3 and r^2S due to the core movement during the algorithm can be neglected.

6.3.1 Rank Adaptivity

Rank adaptivity can be added to the specialized ASD in the same way as for the basic ASD in Section 5.3.5. In particular, as all stack operations, as well as the calculations of $\mathcal{W}_\mu$ and $\left\|\hat{\mathcal{A}}\left(\hat{P}_\mu\left(\hat{\mathcal{A}}^T\left(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\right)\right)\right)\right\|_2^2$ are done on a per-sample basis, it is possible to split the set of samples and use several parallel optimizations in the block-TT format, to obtain a rank adaptive algorithm. The resulting procedure is completely analogous to Algorithm 10, with the difference that the specialized stacks and calculations introduced in this section are used.

6.3.2 Quantities of Interest

Let us note that many statistical properties of the solution are directly available from the TT reconstruction. For example, assuming appropriate orthonormal polynomials are used,

Algorithm 11: Alternating Steepest Descent for UQ Recovery

```
Input: Initial guess $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d \in \mathbb{R}^{n_1 \times \ldots \times n_d}$, measurement operator
       $\hat{\mathcal{A}} : \mathbb{R}^{S \times n_1 \times \ldots \times n_M} \to \mathbb{R}^{S \cdot N}$ and measured values $\boldsymbol{b} \in \mathbb{R}^{S \cdot N}$
Output: Solution $\mathcal{X} = \mathcal{U}_1 \circ \ldots \circ \mathcal{U}_d \in \mathbb{R}^{n_1 \times \ldots \times n_d}$
1  Subroutine Update_core($\mu$)
2      Calculate $\mathcal{W}_\mu$ according to (6.14).
3      Calculate $\left\|\hat{\mathcal{A}}\Big(\hat{P}_\mu\Big(\hat{\mathcal{A}}^T\Big(\hat{\mathcal{A}}(\mathcal{X}) - \boldsymbol{b}\Big)\Big)\Big)\right\|_2^2$ according to (6.15).
4      Caclulate step size as $\beta = \frac{\|\mathcal{W}_\mu\|_F^2}{\left\|\hat{\mathcal{A}}(\hat{P}_\mu(\hat{\mathcal{A}}^T(\hat{\mathcal{A}}(\mathcal{X})-\boldsymbol{b})))\right\|_2^2}$
5      Update $\mathcal{U}_\mu := \mathcal{U}_\mu - \beta \mathcal{W}_\mu$.
6  begin
7      Move core to position 1.
8      Calculate $\mathcal{S}^{\text{Right}}_{\mu,k}$ for $\mu = M+1, \ldots, 2$, according to (6.13).
9      while stopping criteria not fullfilled do
10         for $\mu = 1, \ldots, M+1$ do
11             Update_core($\mu$)
12             if $\mu < M+1$ then
13                 Move core to position $\mu + 1$
14                 Calculate $\mathcal{S}^{\text{LeftIs}}_{\mu,k}$ according to (6.11).
15                 Calculate $\mathcal{S}^{\text{LeftOught}}_{\mu,k}$ according to (6.12).
16         for $\mu = M+1, \ldots, 1$ do
17             Update_core($\mu$)
18             if $\mu > 1$ then
19                 Move core to position $\mu - 1$
20                 Calculate $\mathcal{S}^{\text{Right}}_{\mu,k}$ according to (6.13).
```

the expected value is directly given as

$$\begin{aligned}
\mathbb{E}[u(\boldsymbol{x}, \cdot)] &= \int_\Gamma \tilde{u}(\boldsymbol{x}, \boldsymbol{y}) d\gamma(\boldsymbol{y}) \\
&= \int_\Gamma \mathcal{T} \circ_{M+1} (\boldsymbol{\Phi}(\boldsymbol{x}) \otimes \boldsymbol{\xi}_1(y'_1) \otimes \ldots \otimes \boldsymbol{\xi}_M(y'_M)) d\gamma(\boldsymbol{y}) \\
&= \mathcal{T} \circ_{M+1} (\boldsymbol{\Phi}(\boldsymbol{x}) \otimes \boldsymbol{e}_1 \otimes \ldots \otimes \boldsymbol{e}_1) \\
&= \sum_{j=1}^{S} \mathcal{T}[j, 1, \ldots, 1] \varphi_j(\boldsymbol{x}) \ ,
\end{aligned}$$

which can be obtained from the TT representation at negligible costs. Here, we used that $\int_{\Gamma_\nu} P_k^{(\nu)}(y_\nu) d\gamma_\nu(y_\nu) = \delta_{k,1}$ is non-zero only for the constant polynomial.

Another benefit of the TT representation is that the solution for any given parameter vector $\boldsymbol{y}'$, i.e. any realization of the random variables, can be calculated at significantly reduced costs via the representation (6.5) as

$$\tilde{u}(\boldsymbol{x}, \boldsymbol{y}') = \mathcal{T} \circ_{M+1} \left(\boldsymbol{\Phi}(\boldsymbol{x}) \otimes \boldsymbol{\xi}_1(y_1') \otimes \ldots \otimes \boldsymbol{\xi}_M(y_M')\right) .$$

The computational complexity of these contractions scales as $\mathcal{O}(Sr+Mr^2n)$. For reasonable ranks, this is a drastic cost reduction compared to actually solving the PDE with the corresponding parameters, as done in Monte Carlo sampling.

6.4 Numerical Experiments

This section presents several numerical experiments illustrating the performance of the proposed tensor reconstruction approach for several benchmark problems. In all experiments, we examine the quality of the functional tensor representation of the stochastic solution by comparing the obtained first and second stochastic moments with Monte Carlo simulations using the exact same samples, as well as a Quasi Monte Carlo simulation using Sobol sequence points. In all settings, a reference solution is calculated from at least 256 000 Sobol sequence points. As an additional benchmark, we calculate the error of the functional tensor representation in predicting the solution for new parameter combinations that were *not* used for the reconstruction. More than the stochastic moments, which can also be obtained by MC and QMC methods, this ability to predict the result for new parameters shows the real power of the functional representation.

6.4.1 Implementation Details

In all setups, the domain is given as $D = (0,1)^2$ and a lowest-order conforming Finite Element discretization is employed. As described in Section 6.1.1, we expand the stochastic parameters either in orthogonal Legendre (for uniform random variables) or Hermite (for normal random variables) polynomials. The maximal degree is uniformly fixed to $n = 5$ for the Hermite bases and $n = 13$ for the Legendre bases. For the reconstruction, we employ the block variant of the specialized ASD introduced in the previous section. We divide the samples in $L = 2$ equally large sets plus the control set, which contains 10% of the measurements. The algorithm is stopped, if at the end of one iteration either the current relative residual norm is smaller than 10^{-8}, or if the algorithm failed to find a solution for which the residual norm on the control set decreased by at least a factor of 0.9999 in the last 100 iterations. We limit the maximal allowed rank for each entry of the TT-rank to 50 and additionally prevent excessive rank increases by allowing each entry of the TT-rank to increase only once every 10 iterations. At the beginning of the iteration the accuracy is set to $\epsilon = 10^{-2}$, which is decreased by a factor 0.8 (to a minimum of 10^{-10}) if the relative residual norm did not improve by at least a factor 0.995 during the last ten iterations and if the ranks can increase, i.e. if there was no rank increase in the last ten iterations.

The computation of the FE solutions is performed with the freely available `FEniCS` [178] software packages. For the tensor reconstructions, the `xerus` C++ library (see Appendix A) is used, which includes the complete implementation of our (block) UQ-ASD algorithm.

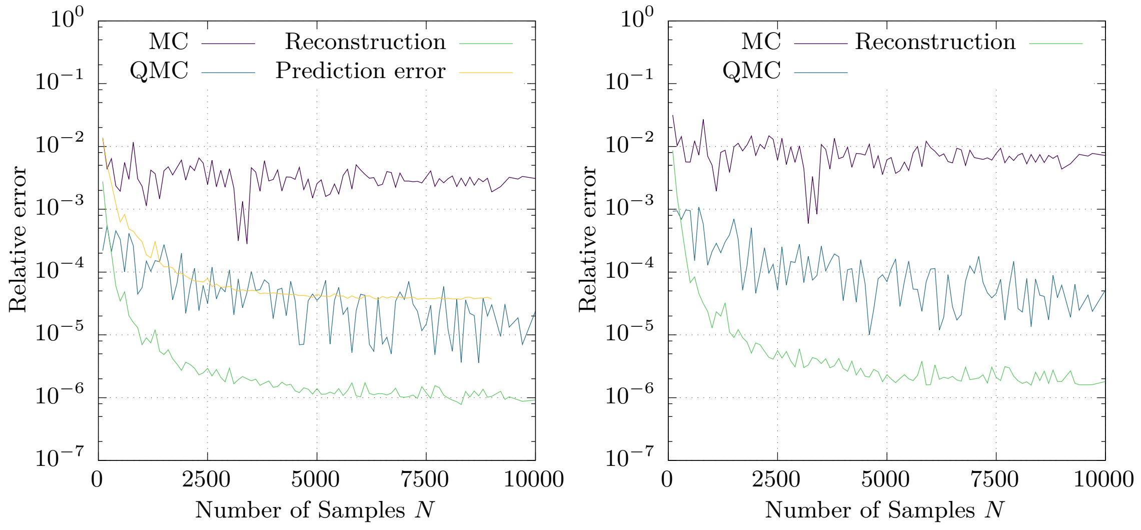


Figure 6.2: Comparison of the reconstruction results for the parametric PDE setting (I) to a MC and QMC simulation. The graphics show the relative error for the first (left) and second (right) stochastic moment for different numbers of samples. The reconstruction and the MC simulation use the exact same samples, which are chosen randomly for each size. Additionally, on the left, the relative error in predicting the solution for 1000 random parameter combinations (which are *not* used for the reconstruction) is given ("Prediction error").

6.4.2 Parametric PDEs

The first class of problems examined are linear second order partial differential equations containing a stochastic field, which is parametrized using a random field expansion (6.3) or (6.4) in independent variables as introduced in Section 6.1.1. For the benchmark problems, we assume that the expansion is given by

$$a_0 := 1, \quad a_m(\boldsymbol{x}) := \alpha_m \cos(2\pi\beta_1(m)x_1)\cos(2\pi\beta_2(m)x_2), \quad m \in \mathbb{N},$$

where $\alpha_m = \frac{9}{10\zeta(2)} m^{-2}$ with the Riemann zeta function ζ. Moreover,

$$\beta_1(m) = m - k(m)(k(m)+1)/2 \quad \text{and} \quad \beta_2(m) = k(m) - \beta_1(m),$$

with $k(m) := \lfloor -1/2 + \sqrt{1/4 + 2m} \rfloor$. This choice corresponds to the "slow decay" experiments in the work of Eigel et al. [158, 171]. In both experiments, the random field expansion is truncated after $M = 20$ terms. The following stationary parametric PDEs with homogeneous Dirichlet boundary conditions are considered:

(I) **Diffusion** (affine)

$$-\boldsymbol{\nabla}(a(\boldsymbol{x},\boldsymbol{y})\boldsymbol{\nabla} u(\boldsymbol{x},\boldsymbol{y})) = 1$$

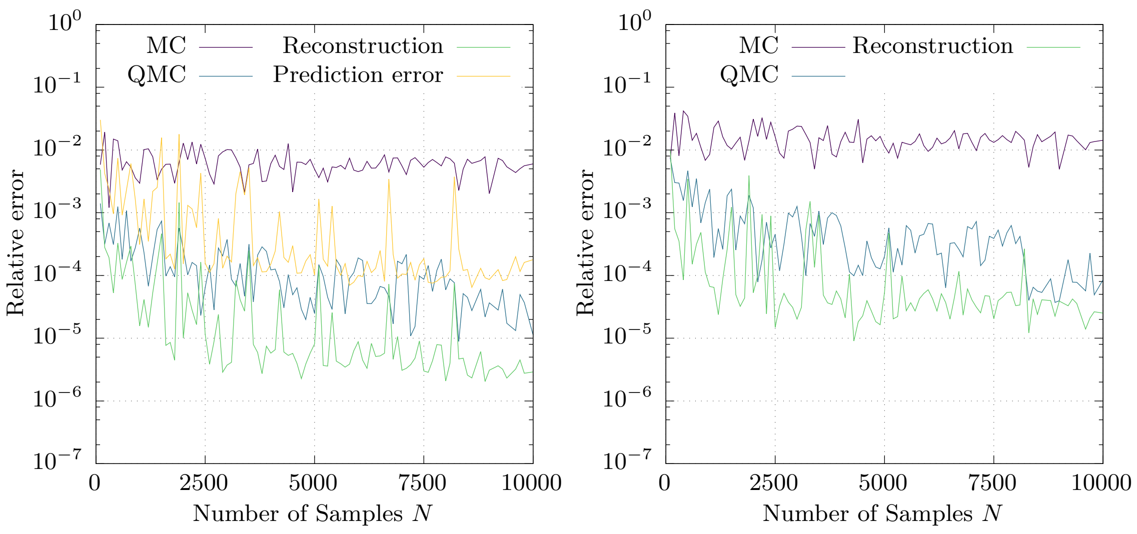


Figure 6.3: Comparison of the reconstruction results for the parametric PDE setting (II) to a MC and QMC simulation. The graphics show the relative error for the first (left) and second (right) stochastic moment for different numbers of samples. The reconstruction and the MC simulation use the exact same samples, which are chosen randomly for each size. Additionally, on the left, the relative error in predicting the solution for 1000 random parameter combinations (which are *not* used for the reconstruction) is given ("Prediction error").

(II) **Diffusion** (lognormal)

$$-\boldsymbol{\nabla}(\exp(a(\boldsymbol{x},\boldsymbol{y}))\boldsymbol{\nabla}u(\boldsymbol{x},\boldsymbol{y})) = 1$$

The results for the two settings are shown in Figures 6.2 and 6.3, respectively. In comparison to the Monte Carlo simulations, using the exact same samples, the approximation of the stochastic moments achieved by the reconstruction is usually better by at least two orders of magnitude in both settings. The results of the reconstruction are also significantly better than the Quasi Monte Carlo simulation, although the latter likewise outperforms the MC simulation. It is noteworthy that the difference in the error between the MC/QMC simulations and the reconstruction is much larger in Setting I than in Setting II, where the QMC simulation partially achieves a similar error as the reconstruction.

For all comparisons between the QMC simulation and the reconstruction or the MC simulation, it is important to note that the reference solution used to calculate the errors is also the result of a QMC simulation using Sobol sequence points. Although there are many more points used to determine the reference solution, this means that it is somewhat biased towards the QMC simulation, as both use the same Sobol sequence. Furthermore, the relatively small variance in the error sometimes observed for the QMC simulation is expected, because the samples for the reconstruction and MC simulation are chosen randomly each time, while the Sobol sequence points are always the same, differing only by the number of samples used.

The relative error in predicting the solution for 1 000 random parameter combinations

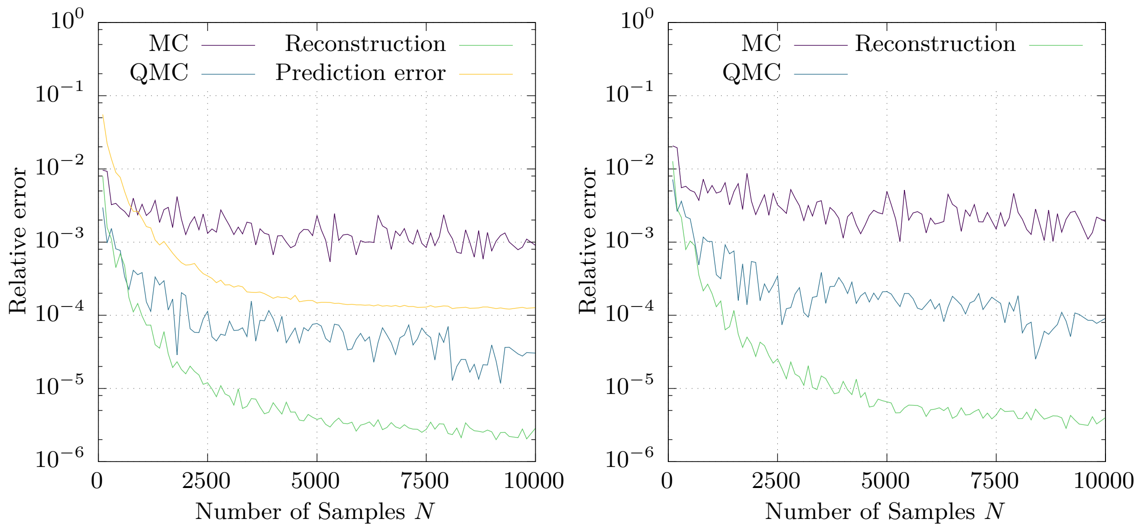


Figure 6.4: Comparison of the reconstruction results to MC and QMC simulations for the cookie setting (III) with fixed radii. The graphics show the relative error for the first (left) and second (right) stochastic moment for different numbers of samples. The reconstruction and the MC simulation use the exact same samples which are chosen randomly for each size. Additionally, on the left, the relative error in predicting the solution for 1000 random parameter combinations (which are *not* used for the reconstruction) is given ("Prediction error").

which were *not* used in the reconstruction initially shows a steep decline with an increased number of samples, but then saturates, showing only a small improvement for further samples. To some extent, this behavior is expected, as the degrees of freedom of the tensor scale as $\mathcal{O}(Mnr^2 + Sr)$, i.e. quadratically in the rank, which means that the number of samples required for successful reconstruction increase at least quadratically as well. Additionally, the accuracy improvement achieved by increasing the rank depends on the relative size of the singular values. Since the relative size declines, the accuracy gained is larger for small ranks. This is apparent in Figure 6.2 for Setting I, where the error barely declines with further samples, which indicates that the limit of the low rank approximability of the solution is reached, in the sense that the part not yet approximated by the low rank solution has (almost) full rank.

6.4.3 Cookie Problems

The second class of problems examined are so-called cookie problems. These are parametric PDEs, in which circular inclusions of the domain with fixed or random size and with different random diffusion coefficients are prescribed. In particular, we consider the following two model settings.

(III) **Cookies with fixed radii.** Let 9 subdomains of $D = (0,1)^2$ be given by discs D_k, $k = 1, \dots, 9$ with fixed radius $R = 1/8$ and centers $\boldsymbol{c} = \begin{pmatrix} i/6 & j/6 \end{pmatrix}^\intercal$ for $i, j \in \{1, 3, 5\}$. The considered problem depends on $\boldsymbol{y} = \begin{pmatrix} y_1 & \dots & y_9 \end{pmatrix}^\intercal$ with $y_k \sim \mathcal{U}(-1, 1)$, has

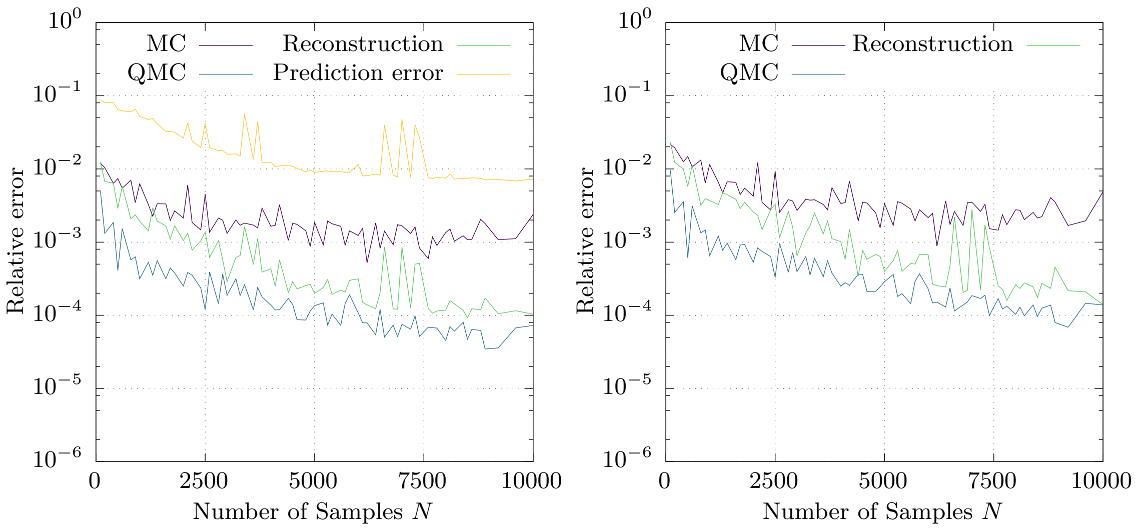


Figure 6.5: Comparison of the reconstruction results to MC and QMC simulations for the cookie setting (IV) with random radii. The graphics show the relative error for the first (left) and second (right) stochastic moment for different numbers of samples. The reconstruction and the MC simulation use the exact same samples which are chosen randomly for each size. Additionally, on the left, the relative error in predicting the solution for 1000 random parameter combinations (which are *not* used for the reconstruction) is given ("Prediction error").

homogeneous Dirichlet boundary conditions and is given by

$$-\boldsymbol{\nabla} \cdot (\kappa(\boldsymbol{x}, \boldsymbol{y}) \boldsymbol{\nabla} u(\boldsymbol{x}, \boldsymbol{y})) = 1,$$

where $\kappa|_{D \setminus \cup_{k=1,\ldots,9} D_k} = 1$ and $\kappa|_{D_k} = y_k$.

(IV) **Cookies with random radii.** The setting is the same as before. However, the problem depends on additional parameters $\boldsymbol{y} = \begin{pmatrix} y_1 & \ldots & y_9 & y_1^* & \ldots & y_9^* \end{pmatrix}^\mathsf{T}$ with $y_k^* \sim \mathcal{U}(-1, 1)$ determining the radii $R_k = (1 + y_k^*/3)/8$ for $k = 1, \ldots, 9$ of the discs D_k.

The results for the two settings are shown in Figures 6.4 and 6.5 respectively. For the setting with fixed radii, the results are largely analogous to Setting I and Setting II. In particular, the error in predicting the stochastic moments is by orders of magnitude smaller than for the MC simulation and also much smaller than for the QMC simulation, except for very small sample sets. The error in predicting the solution for previously unknown random parameter combinations shows the same behavior as for Setting I, i.e. a saturating decline.

The setting with random cookie radii is significantly more difficult, as apparent from the increased errors observed for all three methods. The reconstruction approach is again able to achieve smaller errors in the stochastic moments than the MC simulation, however, the gap is much smaller than for the other settings. Compared to the QMC simulation, the error is even larger in almost all cases. This indicates a rather poor low rank approximability of the solution tensor, which is also recognizable by the much larger error obtained in the prediction of the solution for random parameter combinations.

6.5 Summary and Conclusions

In this chapter, we examined the uncertainty quantification problem for parametric partial differential equations from a tensor reconstruction and statistical learning point of view. The variational Monte Carlo approach introduced in Section 6.2 allows a completely non-intrusive tensor recovery approach for the generation of complete parametric PDE solutions in tensor Hilbert spaces. For the practical realization of this generation, we derived a specialized version of the alternating steepest descent algorithm, which enables the numerical computation of the reconstruction in sub-exponential time with respect to the number of parameters, i.e. with respect to the order of the tensor Hilbert space. Finally, we conducted several numerical experiments examining the quality of the solutions obtained by the reconstruction approach and compared it to standard Monte Carlo and Quasi Monte Carlo methods.

The requirements on the structure of the measurements by the reconstruction approach is minimal, as basically no specific choice of samples is required at all. It can therefore be considered as a direct alternative to Monte Carlo methods, based on principally the same (but possibly much fewer) samples of the parametric solution. In some respects the requirements are even relaxed, as contrary to Monte Carlo simulations, our algorithm can recover a solution even if the samples are drawn from a possibly unknown probability density different from γ. Examining this setting and the involved errors is an interesting topic for future work. In direct comparison in our benchmark problems, the tensor reconstruction often achieves errors in the prediction of the stochastic moments, which are by orders of magnitude smaller than classical Monte Carlo. From another perspective, this means that the reconstruction requires much fewer samples to reach the same accuracy as the Monte Carlo simulation. With exception of the last problem setting, the same is true for the comparison to Quasi Monte Carlo simulations, i.e. the reconstruction usually required significantly fewer samples to reach the same accuracy. Especially for problem settings in which the cost to determine the samples is high and even more so for real world applications, in which samples require a physical measurement, the tensor reconstruction approach should therefore offer a superior use of the available samples, even if only statistical moments like the expected value are desired.

However, the main advantage is that, contrary to Monte Carlo methods where the result is the estimation of some functional (typically the expectation or variance), the proposed reconstruction approach yields a *functional approximation* of the entire solution manifold, from which many quantities of interest can be inexpensively calculated. In this sense, the calculated tensor representation can provide much more information and the comparison with Monte Carlo sampling serves only to illustrate the expected accuracies for these common statistics. The real power of the functional approximation is that the individual solution for *any* parameter combination can be predicted. For reconstructions using 10 000 samples, the relative error of these solutions for random parameter combinations *not* used in the reconstruction is shown in the numerical results to be between $4 \cdot 10^{-5}$ for the diffusion equation with affine coefficient field and $7 \cdot 10^{-3}$ for the cookie problem with random radii. This accuracy is naturally constrained by the accuracy of the discretization introduced in Section 6.1.1 and the low rank approximability of the solution. For the examined parametric linear PDEs with affine and non-affine coefficient fields and the cookie problems, it is apparent (and for the parametric PDEs also known from earlier references) that the solution can indeed be well approximated by tensors with small TT-rank. For many applications, the accuracy reached here should indeed be viable and allow the prediction of solutions for unknown parameter combinations.

Let us finally note that the variational Monte Carlo setting introduced in Section 6.2 is in fact much more general and not limited to the UQ setting. It is the author's belief that further exploration of the relation and inter-connectivity between tensor recovery and machine learning is a promising subject for future work on the topic.

Bibliography

[1] M. Hilbert and P. López. “The world’s technological capacity to store, communicate, and compute information.” In: *science* 332.6025 (2011), pp. 60–65.

[2] *Sensor Sales Keep Hitting New Records But Price Erosion Curbs Growth.* http://www.icinsights.com/data/articles/documents/895.pdf. Accessed: 2018-09-06. IC Insights, 2016.

[3] R. Fujimoto, C. Bock, W. Chen, E. Page, and J. H. Panchal. *Research challenges in modeling and simulation for engineering complex systems.* Springer, 2017.

[4] D. L. Donoho. “Compressed sensing.” In: *IEEE Transactions on information theory* 52.4 (2006), pp. 1289–1306.

[5] E. J. Candès, J. Romberg, and T. Tao. “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information.” In: *IEEE Transactions on information theory* 52.2 (2006), pp. 489–509. arXiv: math/0409186.

[6] M. Lustig, D. Donoho, and J. M. Pauly. “Sparse MRI: The application of compressed sensing for rapid MR imaging.” In: *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine* 58.6 (2007), pp. 1182–1195.

[7] H. Yu and G. Wang. “Compressed sensing based interior tomography.” In: *Physics in medicine & biology* 54.9 (2009), p. 2791.

[8] B. Zhao, J. P. Haldar, C. Brinegar, and Z.-P. Liang. “Low rank matrix recovery for real-time cardiac MRI.” In: *Biomedical Imaging: From Nano to Macro, 2010 IEEE International Symposium on.* IEEE. 2010, pp. 996–999.

[9] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. “Robust face recognition via sparse representation.” In: *IEEE transactions on pattern analysis and machine intelligence* 31.2 (2009), pp. 210–227.

[10] J. F. Gemmeke and B. Cranen. “Using sparse representations for missing data imputation in noise robust speech recognition.” In: *Signal Processing Conference, 2008 16th European.* IEEE. 2008, pp. 1–5.

[11] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al. "Distributed optimization and statistical learning via the alternating direction method of multipliers." In: *Foundations and Trends® in Machine learning* 3.1 (2011), pp. 1–122.

[12] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert. "Quantum state tomography via compressed sensing." In: *Physical review letters* 105.15 (2010), p. 150401. arXiv: 0909.3304.

[13] *Compressive Sensing Resources*. http://dsp.rice.edu/cs/. Rice University.

[14] F. L. Hitchcock. "The expression of a tensor or a polyadic as a sum of products." In: *Journal of Mathematics and Physics* 6.1-4 (1927), pp. 164–189.

[15] L. R. Tucker. "Some mathematical notes on three-mode factor analysis." In: *Psychometrika* 31.3 (1966), pp. 279–311.

[16] W. Hackbusch and S. Kühn. "A new scheme for the tensor representation." In: *Journal of Fourier analysis and applications* 15.5 (2009), pp. 706–722.

[17] I. V. Oseledets and E. E. Tyrtyshnikov. "Breaking the curse of dimensionality, or how to use SVD in many dimensions." In: *SIAM Journal on Scientific Computing* 31.5 (2009), pp. 3744–3759.

[18] L. Grasedyck. "Hierarchical singular value decomposition of tensors." In: *SIAM Journal on Matrix Analysis and Applications* 31.4 (2010), pp. 2029–2054.

[19] A. Uschmajew and B. Vandereycken. "The geometry of algorithms using hierarchical tensors." In: *Linear Algebra and its Applications* 439.EPFL-ARTICLE-197638 (2013), pp. 133–166.

[20] I. V. Oseledets. "Tensor-train decomposition." In: *SIAM Journal on Scientific Computing* 33.5 (2011), pp. 2295–2317.

[21] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac. "Matrix product state representations." In: *arXiv preprint* (2006). arXiv: quant-ph/0608197.

[22] U. Schollwöck. "The density-matrix renormalization group in the age of matrix product states." In: *Annals of Physics* 326.1 (2011), pp. 96–192. arXiv: 1008.3477.

[23] V. Khoromskaia, B. Khoromskij, and R. Schneider. "QTT representation of the Hartree and exchange operators in electronic structure calculations." In: *Computational Methods in Applied Mathematics Comput. Methods Appl. Math.* 11.3 (2011), pp. 327–341.

[24] J. Eisert, M. Cramer, and M. B. Plenio. "Colloquium: Area laws for the entanglement entropy." In: *Reviews of Modern Physics* 82.1 (2010), p. 277. arXiv: 0808.3773.

[25] S. Szalay, M. Pfeffer, V. Murg, G. Barcza, F. Verstraete, R. Schneider, and Ö. Legeza. "Tensor product methods and entanglement optimization for ab initio quantum chemistry." In: *International Journal of Quantum Chemistry* 115.19 (2015), pp. 1342–1391. arXiv: 1412.5829.

[26] B. Khoromskij. "Structured data-sparse approximation to high order tensors arising from the deterministic Boltzmann equation." In: *Mathematics of computation* 76.259 (2007), pp. 1291–1315.

[27] M. Hegland and J. Garcke. "On the numerical solution of the chemical master equation with sums of rank one tensors." In: *ANZIAM Journal* 52 (2011), pp. 628–643.

[28] V. Kazeev, M. Khammash, M. Nip, and C. Schwab. "Direct solution of the chemical master equation using quantized tensor trains." In: *PLoS computational biology* 10.3 (2014), e1003359.

[29] P. Gelß, S. Klus, S. Matera, and C. Schütte. "Nearest-neighbor interaction systems in the tensor-train format." In: *Journal of Computational Physics* 341 (2017), pp. 140–162. arXiv: 1611.03755.

[30] F. Miwakeichi, E. Martınez-Montes, P. A. Valdés-Sosa, N. Nishiyama, H. Mizuhara, and Y. Yamaguchi. "Decomposing EEG data into space–time–frequency components using parallel factor analysis." In: *NeuroImage* 22.3 (2004), pp. 1035–1045.

[31] M. Mørup, L. K. Hansen, J. Parnas, and S. M. Arnfred. *Decomposing the time-frequency representation of EEG using non-negative matrix and multi-way factorization.* Tech. rep. 2006.

[32] M. De Vos, L. De Lathauwer, B. Vanrumste, S. Van Huffel, and W. Van Paesschen. "Canonical decomposition of ictal scalp EEG and accurate source localisation: Principles and simulation study." In: *Computational intelligence and neuroscience* 2007 (2007).

[33] D. Muti and S. Bourennane. "Multidimensional filtering based on a tensor approach." In: *Signal Processing* 85.12 (2005), pp. 2338–2353.

[34] N. D. Sidiropoulos, R. Bro, and G. B. Giannakis. "Parallel factor analysis in sensor array processing." In: *IEEE transactions on Signal Processing* 48.8 (2000), pp. 2377–2388.

[35] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos. "Tensor decomposition for signal processing and machine learning." In: *IEEE Transactions on Signal Processing* 65.13 (2017), pp. 3551–3582. arXiv: 1607.01668.

[36] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov. "Tensorizing neural networks." In: *Advances in Neural Information Processing Systems*. 2015, pp. 442–450. arXiv: 1509.06569.

[37] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky. "Speeding-up convolutional neural networks using fine-tuned cp-decomposition." In: *arXiv preprint* (2014). arXiv: 1412.6553.

[38] E. Stoudenmire and D. J. Schwab. "Supervised learning with tensor networks." In: *Advances in Neural Information Processing Systems*. 2016, pp. 4799–4807. arXiv: 1605.05775.

[39] M. A. O. Vasilescu and D. Terzopoulos. "Multilinear image analysis for facial recognition." In: *Pattern Recognition, 2002. Proceedings. 16th International Conference on*. Vol. 2. IEEE. 2002, pp. 511–514.

[40] M. A. O. Vasilescu and D. Terzopoulos. "Multilinear subspace analysis of image ensembles." In: *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*. Vol. 2. IEEE. 2003, pp. II–93.

[41] A. Shashua and T. Hazan. "Non-negative tensor factorization with applications to statistics and computer vision." In: *Proceedings of the 22nd international conference on Machine learning*. ACM. 2005, pp. 792–799.

[42] J. Liu, P. Musialski, P. Wonka, and J. Ye. "Tensor completion for estimating missing values in visual data." In: *IEEE transactions on pattern analysis and machine intelligence* 35.1 (2013), pp. 208–220.

[43] T. G. Kolda, B. W. Bader, and J. P. Kenny. "Higher-order web link analysis using multilinear algebra." In: *Data Mining, Fifth IEEE International Conference on*. IEEE. 2005, 8–pp.

[44] J. Sun, D. Tao, and C. Faloutsos. "Beyond streams and graphs: dynamic tensor analysis." In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2006, pp. 374–383.

[45] T. G. Kolda and B. W. Bader. "Tensor decompositions and applications." In: *SIAM review* 51.3 (2009), pp. 455–500.

[46] L. Grasedyck, D. Kressner, and C. Tobler. "A literature survey of low-rank tensor approximation techniques." In: *GAMM-Mitteilungen* 36.1 (2013), pp. 53–78.

[47] N. Halko, P.-G. Martinsson, and J. A. Tropp. "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions." In: *SIAM review* 53.2 (2011), pp. 217–288. arXiv: 0909.4061.

[48] B. Recht, M. Fazel, and P. A. Parrilo. “Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization.” In: *SIAM review* 52.3 (2010), pp. 471–501. arXiv: 0706.4138.

[49] E. J. Candès and B. Recht. “Exact matrix completion via convex optimization.” In: *Foundations of Computational mathematics* 9.6 (2009), p. 717. arXiv: 0805.4471.

[50] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup. “Scalable tensor factorizations for incomplete data.” In: *Chemometrics and Intelligent Laboratory Systems* 106.1 (2011), pp. 41–56. arXiv: 1005.2197.

[51] Q. Zhao, L. Zhang, and A. Cichocki. “Bayesian CP factorization of incomplete tensors with automatic rank determination.” In: *IEEE transactions on pattern analysis and machine intelligence* 37.9 (2015), pp. 1751–1763. arXiv: 1401.6497.

[52] L. Karlsson, D. Kressner, and A. Uschmajew. “Parallel algorithms for tensor completion in the CP format.” In: *Parallel Computing* 57 (2016), pp. 222–234.

[53] T. Yokota, Q. Zhao, and A. Cichocki. “Smooth PARAFAC decomposition for tensor completion.” In: *IEEE Transactions on Signal Processing* 64.20 (2016), pp. 5423–5436. arXiv: 1505.06611.

[54] H. Rauhut and Ž. Stojanac. “Tensor theta norms and low rank recovery.” In: *arXiv preprint* (2015). arXiv: 1505.05175.

[55] S. Gandy, B. Recht, and I. Yamada. “Tensor completion and low-n-rank tensor recovery via convex optimization.” In: *Inverse Problems* 27.2 (2011), p. 025010.

[56] H. Rauhut, R. Schneider, and Z. Stojanac. “Low rank tensor recovery via iterative hard thresholding.” In: *Proc. 10th International Conference on Sampling Theory and Applications.* 2013.

[57] D. Kressner, M. Steinlechner, and B. Vandereycken. “Low-rank tensor completion by Riemannian optimization.” In: *BIT Numerical Mathematics* 54.2 (2014), pp. 447–468.

[58] M. Filipović and A. Jukić. “Tucker factorization with missing data with application to low-n-rank tensor completion.” In: *Multidimensional systems and signal processing* 26.3 (2015), pp. 677–692.

[59] L. Grasedyck, M. Kluge, and S. Krämer. “Alternating directions fitting (ADF) of hierarchical low rank tensors.” In: *Preprint* 149 (2013).

[60] L. Grasedyck, M. Kluge, and S. Krämer.
"Variants of alternating least squares tensor completion in the tensor train format."
In: *SIAM Journal on Scientific Computing* 37.5 (2015), A2424–A2450.

[61] H. Rauhut, R. Schneider, and Ž. Stojanac.
"Tensor completion in hierarchical tensor representations."
In: *Compressed Sensing and its Applications*. Springer, 2015, pp. 419–450.
arXiv: 1404.3905.

[62] H. Kasai and B. Mishra.
"Low-rank tensor completion: a Riemannian manifold preconditioning approach."
In: *International Conference on Machine Learning*. 2016, pp. 1012–1021.
arXiv: 1605.08257.

[63] M. Steinlechner.
"Riemannian optimization for high-dimensional tensor completion."
In: *SIAM Journal on Scientific Computing* 38.5 (2016), S461–S484.

[64] M. M. Steinlechner. "Riemannian optimization for solving high-dimensional problems with low-rank tensor structure." PhD thesis. EPFL, 2016.

[65] H. Rauhut, R. Schneider, and Ž. Stojanac.
"Low rank tensor recovery via iterative hard thresholding."
In: *Linear Algebra and its Applications* 523 (2017), pp. 220–262.
arXiv: 1602.05217.

[66] L. Grasedyck and S. Krämer. "Stable ALS Approximation in the TT-Format for Rank-Adaptive Tensor Completion." In: *arXiv preprint* (2017). arXiv: 1701.08045.

[67] M. Ashraphijuo and X. Wang. "Characterization of deterministic and probabilistic sampling patterns for finite completability of low tensor-train rank tensor."
In: *arXiv preprint* (2017). arXiv: 1703.07698.

[68] A. Uschmajew. "Zur Theorie der Niedrigrangapproximation in Tensorprodukten von Hilberträumen." PhD thesis. 2013.

[69] M. Reed and B. Simon.
Methods of modern mathematical physics. vol. 1. Functional analysis.
Academic New York, 1980.

[70] W. Hackbusch. *Tensor spaces and numerical tensor calculus*. Vol. 42.
Springer, 2012.

[71] G. H. Golub and C. F. Van Loan. *Matrix computations*. Vol. 3. JHU Press, 2012.

[72] C. Eckart and G. Young.
"The approximation of one matrix by another of lower rank."
In: *Psychometrika* 1.3 (1936), pp. 211–218.

[73] G. W. Stewart. "On the early history of the singular value decomposition."
In: *SIAM review* 35.4 (1993), pp. 551–566.

[74] R. A. Horn and C. R. Johnson. *Topics in matrix analysis.* Cambridge University Presss, Cambridge, 1991.

[75] M. E. Hochstenbach and L. Reichel. "Subspace-restricted singular value decompositions for linear discrete ill-posed problems." In: *Journal of Computational and Applied Mathematics* 235.4 (2010), pp. 1053–1064.

[76] L. N. Trefethen and D. Bau III. *Numerical linear algebra.* Vol. 50. Siam, 1997.

[77] U. Helmke and J. B. Moore. *Optimization and dynamical systems.* Springer, 1994.

[78] J. Håstad. "Tensor rank is NP-complete." In: *Journal of Algorithms* 11.4 (1990), pp. 644–654.

[79] V. De Silva and L.-H. Lim. "Tensor rank and the ill-posedness of the best low-rank approximation problem." In: *SIAM Journal on Matrix Analysis and Applications* 30.3 (2008), pp. 1084–1127. arXiv: math/0607647.

[80] N. K. M. Faber, R. Bro, and P. K. Hopke. "Recent developments in CANDECOMP/PARAFAC algorithms: a critical review." In: *Chemometrics and Intelligent Laboratory Systems* 65.1 (2003), pp. 119–137.

[81] B. C. Mitchell and D. S. Burdick. "Slowly converging PARAFAC sequences: swamps and two-factor degeneracies." In: *Journal of Chemometrics* 8.2 (1994), pp. 155–168.

[82] T. G. Kolda. "Orthogonal tensor decompositions." In: *SIAM Journal on Matrix Analysis and Applications* 23.1 (2001), pp. 243–255.

[83] L. De Lathauwer, B. De Moor, and J. Vandewalle. "A multilinear singular value decomposition." In: *SIAM journal on Matrix Analysis and Applications* 21.4 (2000), pp. 1253–1278.

[84] C. J. Hillar and L.-H. Lim. "Most tensor problems are NP-hard." In: *Journal of the ACM (JACM)* 60.6 (2013), p. 45. arXiv: 0911.1393.

[85] S. R. White. "Density matrix formulation for quantum renormalization groups." In: *Physical review letters* 69.19 (1992), p. 2863.

[86] I. Oseledets and E. Tyrtyshnikov. "TT-cross approximation for multidimensional arrays." In: *Linear Algebra and its Applications* 432.1 (2010), pp. 70–88.

[87] S. Holtz, T. Rohwedder, and R. Schneider. "On manifolds of tensors of fixed TT-rank." In: *Numerische Mathematik* 120.4 (2012), pp. 701–731.

[88] C. Lubich, I. V. Oseledets, and B. Vandereycken. "Time integration of tensor trains." In: *SIAM Journal on Numerical Analysis* 53.2 (2015), pp. 917–941. arXiv: 1407.2042.

[89] W. Hackbusch and R. Schneider.
"Tensor spaces and hierarchical tensor representations."
In: *Extraction of Quantifiable Information from Complex Systems*. Springer, 2014, pp. 237–261.

[90] L. Grasedyck and W. Hackbusch.
"An introduction to hierarchical (H-) rank and TT-rank of tensors with examples."
In: *Computational Methods in Applied Mathematics Comput. Methods Appl. Math.* 11.3 (2011), pp. 291–304.

[91] F. Verstraete, V. Murg, and J. I. Cirac.
"Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems."
In: *Advances in Physics* 57.2 (2008), pp. 143–224. arXiv: 0907.2796.

[92] G. Vidal. "Class of quantum many-body states that can be efficiently simulated."
In: *Physical review letters* 101.11 (2008), p. 110501. arXiv: quant-ph/0610099.

[93] Y.-Y. Shi, L.-M. Duan, and G. Vidal.
"Classical simulation of quantum many-body systems with a tree tensor network."
In: *Physical review a* 74.2 (2006), p. 022320. arXiv: quant-ph/0511070.

[94] J. M. Landsberg, Y. Qi, and K. Ye. "On the geometry of tensor network states."
In: *arXiv preprint* (2011). arXiv: 1105.4449.

[95] B. Huber, R. Schneider, and S. Wolf.
"A Randomized Tensor Train Singular Value Decomposition."
In: *Compressed Sensing and its Applications*. Springer, 2017, pp. 261–290. arXiv: 1710.08513.

[96] J. H. d. M. Goulart and G. Favier. "An iterative hard thresholding algorithm with improved convergence for low-rank tensor recovery."
In: *Signal Processing Conference (EUSIPCO), 2015 23rd European*. IEEE. 2015, pp. 1701–1705.

[97] S. Holtz, T. Rohwedder, and R. Schneider.
"The alternating linear scheme for tensor optimization in the tensor train format."
In: *SIAM Journal on Scientific Computing* 34.2 (2012), A683–A713.

[98] M. Espig, W. Hackbusch, and A. Khachatryan. "On the convergence of alternating least squares optimisation in tensor format representations."
In: *arXiv preprint* (2015). arXiv: 1506.00062.

[99] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin.
"A theory of pseudoskeleton approximations."
In: *Linear algebra and its applications* 261.1-3 (1997), pp. 1–21.

[100] E. Tyrtyshnikov. "Incomplete cross approximation in the mosaic-skeleton method."
In: *Computing* 64.4 (2000), pp. 367–380.

[101] M. Bebendorf. "Approximation of boundary element matrices."
In: *Numerische Mathematik* 86.4 (2000), pp. 565–589.

[102] M. Bebendorf and S. Rjasanow.
"Adaptive low-rank approximation of collocation matrices."
In: *Computing* 70.1 (2003), pp. 1–24.

[103] I. V. Oseledets, D. Savostianov, and E. E. Tyrtyshnikov.
"Tucker dimensionality reduction of three-dimensional arrays in linear time."
In: *SIAM Journal on Matrix Analysis and Applications* 30.3 (2008), pp. 939–956.

[104] M. Bebendorf. "Adaptive cross approximation of multivariate functions."
In: *Constructive approximation* 34.2 (2011), pp. 149–179.

[105] J. Ballani, L. Grasedyck, and M. Kluge.
"Black box approximation of tensors in hierarchical Tucker format."
In: *Linear algebra and its applications* 438.2 (2013), pp. 639–657.

[106] S. Wolf. "Low Rank Tensor Recovery in the Tensor Train Representation."
MA thesis. Freie Universität Berlin, 2014.

[107] J. Bennett, S. Lanning, et al. "The netflix prize."
In: *Proceedings of KDD cup and workshop*. Vol. 2007. New York, NY, USA. 2007, p. 35.

[108] R. M. Bell and Y. Koren. "Lessons from the Netflix prize challenge."
In: *Acm Sigkdd Explorations Newsletter* 9.2 (2007), pp. 75–79.

[109] J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert.
"Low-rank matrix factorization with attributes." In: *arXiv preprint* (2006).
arXiv: cs/0611124.

[110] Y. Amit, M. Fink, N. Srebro, and S. Ullman.
"Uncovering shared structures in multiclass classification."
In: *Proceedings of the 24th international conference on Machine learning*.
ACM. 2007, pp. 17–24.

[111] A. Argyriou, T. Evgeniou, and M. Pontil. "Multi-task feature learning."
In: (2007), pp. 41–48.

[112] C. Tomasi and T. Kanade. "Shape and motion from image streams under orthography: a factorization method."
In: *International Journal of Computer Vision* 9.2 (1992), pp. 137–154.

[113] A. Singer. "A remark on global positioning from local distances."
In: *Proceedings of the National Academy of Sciences* 105.28 (2008), pp. 9507–9511.

[114] A. Y. Alfakih, A. Khandani, and H. Wolkowicz. "Solving Euclidean distance matrix completion problems via semidefinite programming."
In: *Computational optimization and applications* 12.1-3 (1999), pp. 13–30.

[115] B. K. Natarajan. "Sparse approximate solutions to linear systems."
In: *SIAM journal on computing* 24.2 (1995), pp. 227–234.

[116] R. H. Keshavan, A. Montanari, and S. Oh. "Matrix completion from a few entries." In: *IEEE transactions on information theory* 56.6 (2010), pp. 2980–2998. arXiv: 0901.3150.

[117] J.-F. Cai, E. J. Candès, and Z. Shen. "A singular value thresholding algorithm for matrix completion." In: *SIAM Journal on Optimization* 20.4 (2010), pp. 1956–1982. arXiv: 0810.3286.

[118] S. Foucart and H. Rauhut. *A mathematical introduction to compressive sensing.* Vol. 1. 3. Birkhäuser Basel, 2013.

[119] M. Fazel. "Matrix rank minimization with applications." PhD thesis. PhD thesis, Stanford University, 2002.

[120] M. Mesbahi and G. P. Papavassilopoulos. "On the rank minimization problem over a positive semidefinite linear matrix inequality." In: *IEEE Transactions on Automatic Control* 42.2 (1997), pp. 239–243.

[121] M. Fazel, H. Hindi, and S. P. Boyd. "A rank minimization heuristic with application to minimum order system approximation." In: *American Control Conference, 2001. Proceedings of the 2001.* Vol. 6. IEEE. 2001, pp. 4734–4739.

[122] C. R. Johnson. "Matrix completion problems: a survey." In: *Matrix theory and applications.* Vol. 40. Providence, RI. 1990, pp. 171–198.

[123] K. R. Gabriel and S. Zamir. "Lower rank approximation of matrices by least squares with any choice of weights." In: *Technometrics* 21.4 (1979), pp. 489–498.

[124] H. J. Woerdeman. "Minimal rank completions for block matrices." In: *Linear Algebra and its Applications* 121 (1989), pp. 105–122.

[125] E. J. Candès and T. Tao. "Decoding by linear programming." In: *IEEE transactions on information theory* 51.12 (2005), pp. 4203–4215. arXiv: math/0502327.

[126] E. J. Candès and Y. Plan. "Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements." In: *IEEE Transactions on Information Theory* 57.4 (2011), pp. 2342–2359. arXiv: 1001.0339.

[127] E. J. Candès and T. Tao. "The power of convex relaxation: Near-optimal matrix completion." In: *IEEE Transactions on Information Theory* 56.5 (2010), pp. 2053–2080. arXiv: 0903.1476.

[128] B. Recht. "A simpler approach to matrix completion." In: *Journal of Machine Learning Research* 12.Dec (2011), pp. 3413–3430. arXiv: 0910.0651.

[129] E. J. Candès and Y. Plan. "Matrix completion with noise." In: *Proceedings of the IEEE* 98.6 (2010), pp. 925–936. arXiv: 0903.3131.

[130] K. Toh, R. Tütüncü, and M. Todd.
SDPT3–a Matlab software package for semidefinite-quadratic-linear programming.

[131] P. Jain, R. Meka, and I. S. Dhillon.
"Guaranteed rank minimization via singular value projection."
In: *Advances in Neural Information Processing Systems*. 2010, pp. 937–945.
arXiv: 0909.5457.

[132] J. Tanner and K. Wei.
"Normalized iterative hard thresholding for matrix completion."
In: *SIAM Journal on Scientific Computing* 35.5 (2013), S104–S125.

[133] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin.
"An iterative regularization method for total variation-based image restoration."
In: *Multiscale Modeling & Simulation* 4.2 (2005), pp. 460–489.

[134] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. "Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing."
In: *SIAM Journal on Imaging sciences* 1.1 (2008), pp. 143–168.

[135] P. Jain, P. Netrapalli, and S. Sanghavi.
"Low-rank matrix completion using alternating minimization."
In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing.* ACM. 2013, pp. 665–674. arXiv: 1212.0467.

[136] R. Tomioka, T. Suzuki, K. Hayashi, and H. Kashima.
"Statistical performance of convex tensor decomposition."
In: *Advances in Neural Information Processing Systems*. 2011, pp. 972–980.

[137] N. Kreimer and M. D. Sacchi.
"Nuclear norm minimization and tensor completion in exploration seismology."
In: *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. 2013, pp. 4275–4279.

[138] M. Signoretto, L. De Lathauwer, and J. A. Suykens.
"Nuclear norms for tensors and their use for convex multilinear estimation."
In: *Submitted to Linear Algebra and Its Applications* 43 (2010).

[139] H. N. Phien, H. D. Tuan, J. A. Bengua, and M. N. Do.
"Efficient tensor completion: Low-rank tensor train." In: *arXiv preprint* (2016).
arXiv: 1601.01083.

[140] C. Mu, B. Huang, J. Wright, and D. Goldfarb.
"Square deal: Lower bounds and improved relaxations for tensor recovery."
In: *International Conference on Machine Learning*. 2014, pp. 73–81.
arXiv: 1307.5870.

[141] B. Romera-Paredes and M. Pontil.
"A new convex relaxation for tensor completion."
In: *Advances in Neural Information Processing Systems*. 2013, pp. 2967–2975.
arXiv: 1307.4653.

[142] S. Oymak, A. Jalali, M. Fazel, Y. C. Eldar, and B. Hassibi. “Simultaneously Structured Models With Application to Sparse and Low-Rank Matrices.” In: *IEEE Trans. Information Theory* 61.5 (2015), pp. 2886–2908. arXiv: 1212.3753.

[143] M. Yuan and C.-H. Zhang. “Incoherent tensor norms and their applications in higher order tensor completion.” In: *IEEE Transactions on Information Theory* 63.10 (2017), pp. 6753–6766. arXiv: 1606.03504.

[144] T. Blumensath and M. E. Davies. “Iterative hard thresholding for compressed sensing.” In: *Applied and computational harmonic analysis* 27.3 (2009), pp. 265–274. arXiv: 0805.0510.

[145] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds.* Princeton University Press, 2009.

[146] S. V. Dolgov, B. N. Khoromskij, I. V. Oseledets, and D. V. Savostyanov. “Computation of extreme eigenvalues in higher dimensions using block tensor train format.” In: *Computer Physics Communications* 185.4 (2014), pp. 1207–1216. arXiv: 1306.2269.

[147] M. Eigel, J. Neumann, R. Schneider, and S. Wolf. “Non-intrusive Tensor Reconstruction for High-Dimensional Random PDEs.” In: *Computational Methods in Applied Mathematics* (2018).

[148] M. Eigel, R. Schneider, P. Trunschke, and S. Wolf. “Variational Monte Carlo — Bridging concepts of machine learning and high dimensional partial differential equations.”

[149] H. N. Najm. “Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics.” In: *Annual review of fluid mechanics* 41 (2009), pp. 35–52.

[150] S. Wojtkiewicz, M. Eldred, R. Field Jr, A. Urbina, and J. Red-Horse. “Uncertainty quantification in large computational engineering models.” In: *19th AIAA Applied Aerodynamics Conference.* 2001, p. 1455.

[151] M. B. Giles. “Multilevel monte carlo methods.” In: *Acta Numerica* 24 (2015), pp. 259–328.

[152] K. A. Cliffe, M. B. Giles, R. Scheichl, and A. L. Teckentrup. “Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients.” In: *Computing and Visualization in Science* 14.1 (2011), p. 3.

[153] F. Y. Kuo, C. Schwab, and I. H. Sloan. “Quasi-Monte Carlo finite element methods for a class of elliptic partial differential equations with random coefficients.” In: *SIAM Journal on Numerical Analysis* 50.6 (2012), pp. 3351–3374.

[154] I. Babuška, F. Nobile, and R. Tempone. “A stochastic collocation method for elliptic partial differential equations with random input data.” In: *SIAM J. Numer. Anal.* 45.3 (2007), pp. 1005–1034.

[155] F. Nobile, R. Tempone, and C. G. Webster. “A sparse grid stochastic collocation method for partial differential equations with random input data.” In: *SIAM Journal on Numerical Analysis* 46.5 (2008), pp. 2309–2345.

[156] I. Babuška, R. Tempone, and G. E. Zouraris. “Solving elliptic boundary value problems with uncertain coefficients by the finite element method: the stochastic formulation.” In: *Computer methods in applied mechanics and engineering* 194.12-16 (2005), pp. 1251–1294.

[157] J. Giesselmann, F. Meyer, and C. Rohde. “A posteriori error analysis for random scalar conservation laws using the Stochastic Galerkin method.” In: *arXiv preprint* (2017). arXiv: 1709.04351.

[158] M. Eigel, C. J. Gittelson, C. Schwab, and E. Zander. “Adaptive stochastic galerkin FEM.” In: *Computer Methods in Applied Mechanics and Engineering* 270 (2014), pp. 247–269.

[159] B. N. Khoromskij and C. Schwab. “Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs.” In: *SIAM Journal on Scientific Computing* 33.1 (2011), pp. 364–385.

[160] H. G. Matthies, A. Litvinenko, O. Pajonk, B. V. Rosić, and E. Zander. “Parametric and uncertainty computations with tensor product representations.” In: *Uncertainty Quantification in Scientific Computing.* Springer, 2012, pp. 139–150.

[161] M. Espig, W. Hackbusch, A. Litvinenko, H. G. Matthies, and P. Wähnert. “Efficient low-rank approximation of the stochastic Galerkin matrix in tensor formats.” In: *Computers & Mathematics with Applications* 67.4 (2014), pp. 818–829.

[162] B. N. Khoromskij and I. Oseledets. “Quantics-TT collocation approximation of parameter-dependent and stochastic elliptic PDEs.” In: *Computational Methods in Applied Mathematics Comput. Methods Appl. Math.* 10.4 (2010), pp. 376–394.

[163] S. Dolgov, B. N. Khoromskij, A. Litvinenko, and H. G. Matthies. “Computation of the Response Surface in the Tensor Train data format.” 2014.

[164] M. Bachmayr, A. Cohen, and W. Dahmen. “Parametric PDEs: sparse or low-rank approximations?” In: *IMA Journal of Numerical Analysis* (2016). arXiv: 1607.04444.

[165] M. Eigel, M. Pfeffer, and R. Schneider. “Adaptive stochastic Galerkin FEM with hierarchical tensor representations.” In: *Numerische Mathematik* 136.3 (2017), pp. 765–803.

[166] M. Espig, W. Hackbusch, A. Litvinenko, H. G. Matthies, and E. Zander. "Efficient analysis of high dimensional data in tensor formats." In: *Sparse Grids and Applications*. Springer, 2012, pp. 31–56.

[167] S. Dolgov, B. N. Khoromskij, A. Litvinenko, and H. G. Matthies. "Polynomial Chaos Expansion of random coefficients and the solution of stochastic partial differential equations in the Tensor Train format." In: *SIAM/ASA Journal on Uncertainty Quantification* 3.1 (2015), pp. 1109–1135. arXiv: 1503.03210.

[168] S. Dolgov and R. Scheichl. "A hybrid Alternating Least Squares–TT Cross algorithm for parametric PDEs." In: *arXiv preprint* (2017). arXiv: 1707.04562.

[169] C. Schwab and C. J. Gittelson. "Sparse tensor discretizations of high-dimensional parametric and stochastic PDEs." In: *Acta Numerica* 20 (2011), pp. 291–467.

[170] G. J. Lord, C. E. Powell, and T. Shardlow. *An Introduction to Computational Stochastic PDEs.* Cambridge Texts in Applied Mathematics. Cambridge University Press, New York, 2014, pp. xii+503.

[171] M. Eigel, C. J. Gittelson, C. Schwab, and E. Zander. "A convergent adaptive stochastic Galerkin finite element method with quasi-optimal spatial meshes." In: *ESAIM: Mathematical Modelling and Numerical Analysis* 49.5 (2015), pp. 1367–1398.

[172] M. Eigel and C. Merdon. "Local equilibration error estimators for guaranteed error control in adaptive stochastic higher-order Galerkin FEM." In: *WIAS Preprint* (2014).

[173] M. Loève. *Probability Theory. I.* Fourth. Graduate Texts in Mathematics, Vol. 45. Springer-Verlag, New York-Heidelberg, 1977, pp. xvii+425.

[174] C. Schwab and R. A. Todor. "Karhunen–Loève approximation of random fields by generalized fast multipole methods." In: *Journal of Computational Physics* 217.1 (2006), pp. 100–122.

[175] F. Cucker and S. Smale. "On the mathematical foundations of learning." In: *Bulletin of the American mathematical society* 39.1 (2002), pp. 1–49.

[176] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* http://www.deeplearningbook.org. MIT Press, 2016.

[177] J. L. Macdonald. "Image classification with wavelet and shearlet based scattering transforms." MA thesis. Technische Universität Berlin, 2017.

[178] M. S. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, and G. N. Wells. "The FEniCS Project Version 1.5." In: *Archive of Numerical Software* 3.100 (2015).

[179] Free Software Foundation.
GNU Affero General Public License Version 3 (AGPL-3.0).

[180] L. S. Blackford, A. Petitet, R. Pozo, K. Remington, R. C. Whaley, J. Demmel, J. Dongarra, I. Duff, S. Hammarling, G. Henry, et al.
"An updated set of basic linear algebra subprograms (BLAS)."
In: *ACM Transactions on Mathematical Software* 28.2 (2002), pp. 135–151.

[181] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen.
LAPACK Users' Guide. Third.
Philadelphia, PA: Society for Industrial and Applied Mathematics, 1999.

[182] T. A. Davis et al. *Suitesparse: a suite of sparse matrix software.*
`http://www.suitesparse.com`. 2015.

[183] W. S. Zhang Xianyi Wang Qian et al. *OpenBLAS - An optimized BLAS library.*

A The xerus Tensor Library

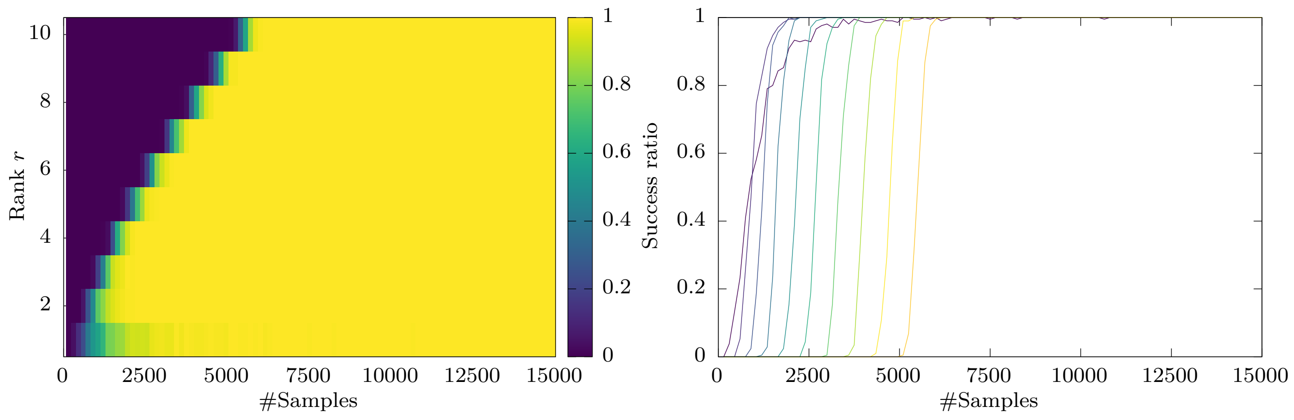
In preparation of this thesis and in collaboration with his college Benjamin Huber, the author designed and implemented a high performance C++ tensor library named `xerus`. The rationale of writing this library was, on the one hand, to have a solid foundation for efficient implementations of the algorithms developed and used in this work. On the other hand, having a self-contained library, which includes all common routines, is a great way to share those algorithms with fellow researches and the general public and allows them to easily modify and extend this work. With this goal in mind, the complete source code is released under the AGPLv3 free software license [179]. The source code, as well as an extensive documentation, can be found on the website `http://libxerus.org`. Today, `xerus` is used by several researchers from different working groups and we are confident that the development of the library will also continue in the future.

In its current state, the library implements classes for dense and sparse tensors of arbitrary orders, general tensors Networks and a specialized class for the TT-format. For both types of tensors and the TT-format, all the elementary operations mentioned in Chapter 3 are implemented. Wherever possible, the efficient algorithms with non-exponential scaling mentioned in Chapter 3 are used. As a means to obtain a TT decomposition, the library contains implementations of the deterministic TT-SVD, as well as the randomized variants presented in Chapter 4. Furthermore, several optimization algorithms for low rank TT-tensors are available in `xerus`, including all variants of the alternating steepest descent (ASD) presented in Chapter 5 and Chapter 6, the alternating least squares (ALS) algorithm and several Riemannian optimization algorithms. For an exhaustive list, we refer to the manual on `http://libxerus.org`.

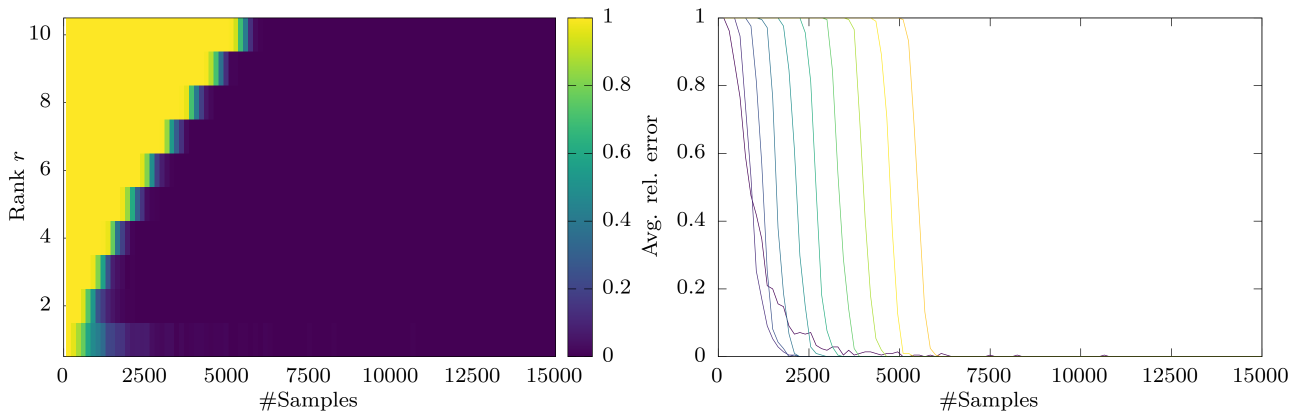
To achieve the best performance, the library directly calls the basic linear algebra subprograms (BLAS) [180] and the Linear Algebra Package (LAPACK) [181], as well as routines from “SuiteSparse” [182]. For the particular calculations presented in this thesis, the “OpenBLAS” [183] implementation of the BLAS is used.

B Complete Numerical Results of Chapter 5

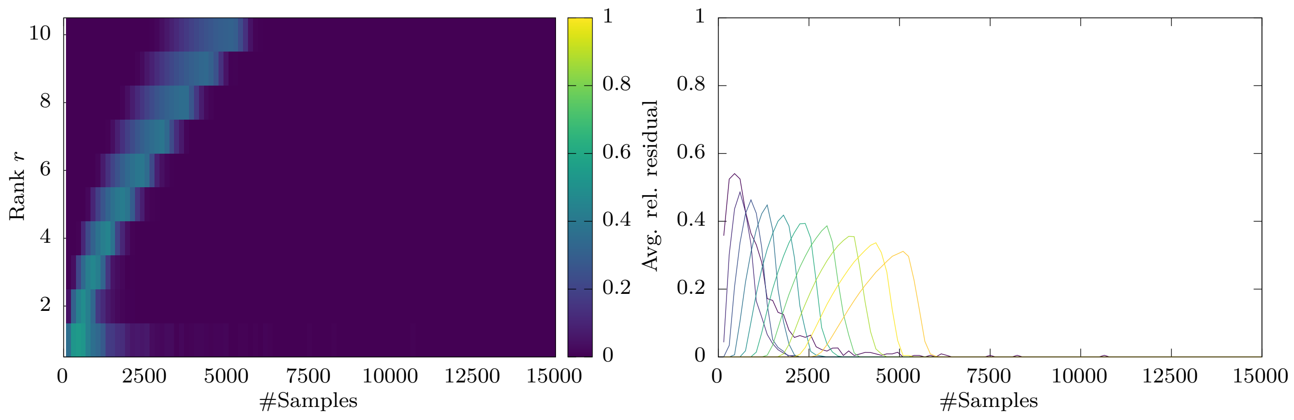
This appendix provides the complete numerical results for the low rank tensor recovery setting described in Chapter 5. The detailed description of all experiments can be found in Section 5.4. The performance of the *alternating steepest descent (ASD)* algorithm for tensor recovery of exact low rank tensors is shown in Figures B.1, B.2 (varying rank) and B.7, B.8 (varying order) for the noiseless and Figures B.3, B.4 (varying rank) and B.9, B.10 (varying order) for the noisy setting. Figures B.5 and B.6 show the results for noiseless recovery of tensors with approximately quadratically decreasing singular values. Figures B.11 and B.12 show the performance of the rank adaptive block-ASD algorithm for tensor recovery of exact low rank tensors using rank-one samples in the noiseless and noisy setting.



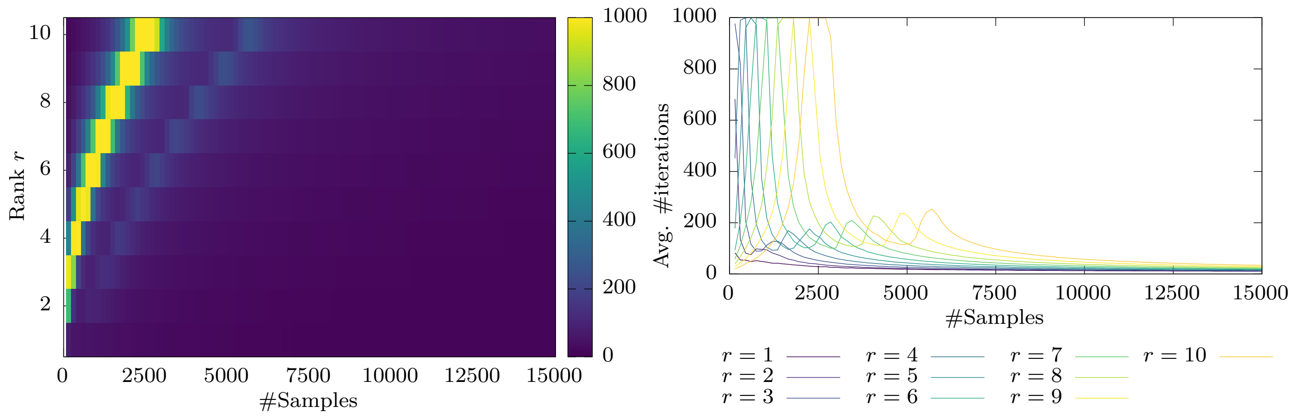
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

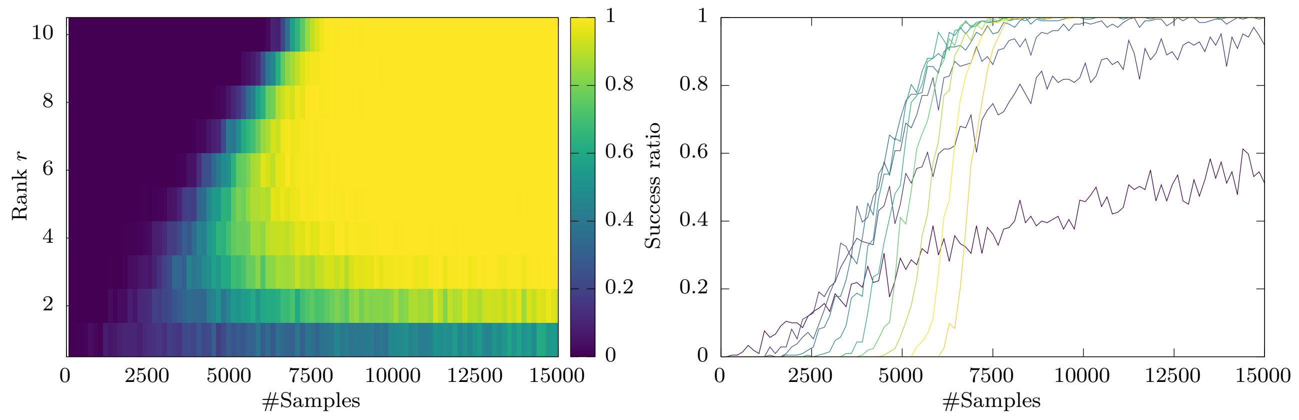


(c) Average relative residual of the solution found by the algorithm.

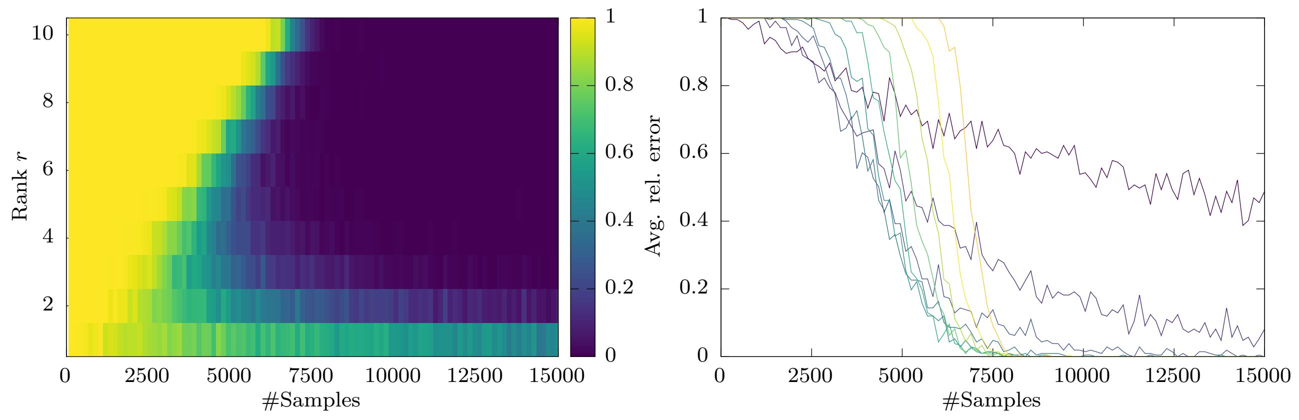


(d) Average number of iterations performed by the algorithm.

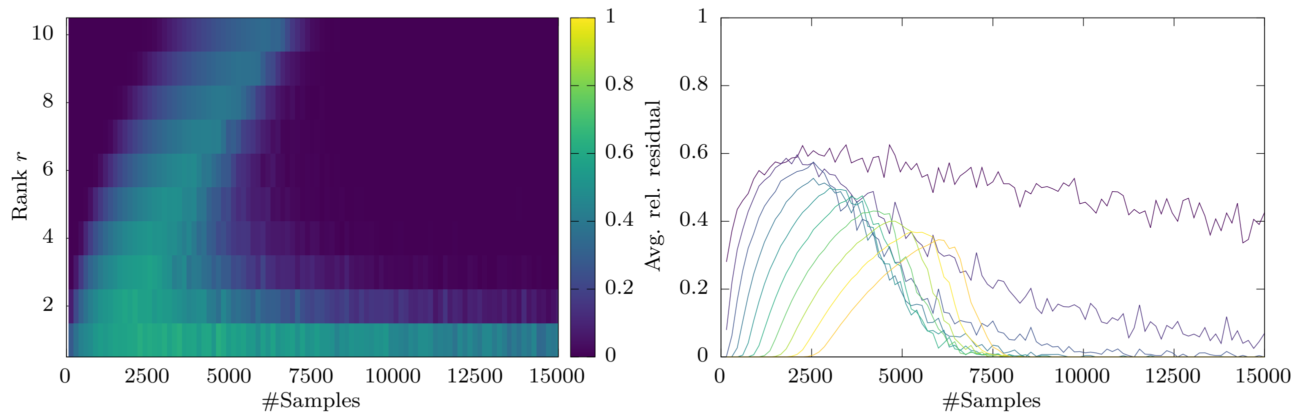
Figure B.1: Numerical results for low rank tensor recovery using the ASD algorithm with rank one samples. The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and varying uniform rank r.



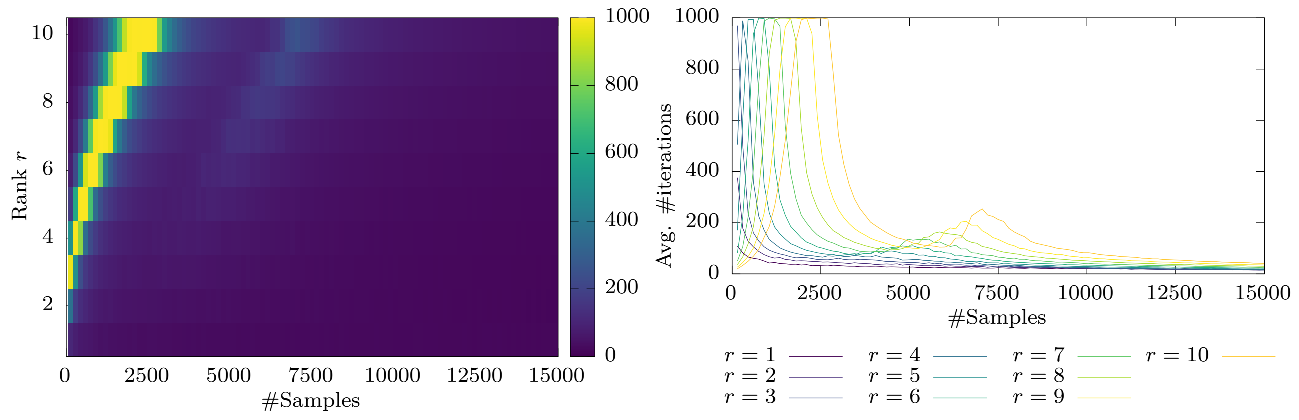
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

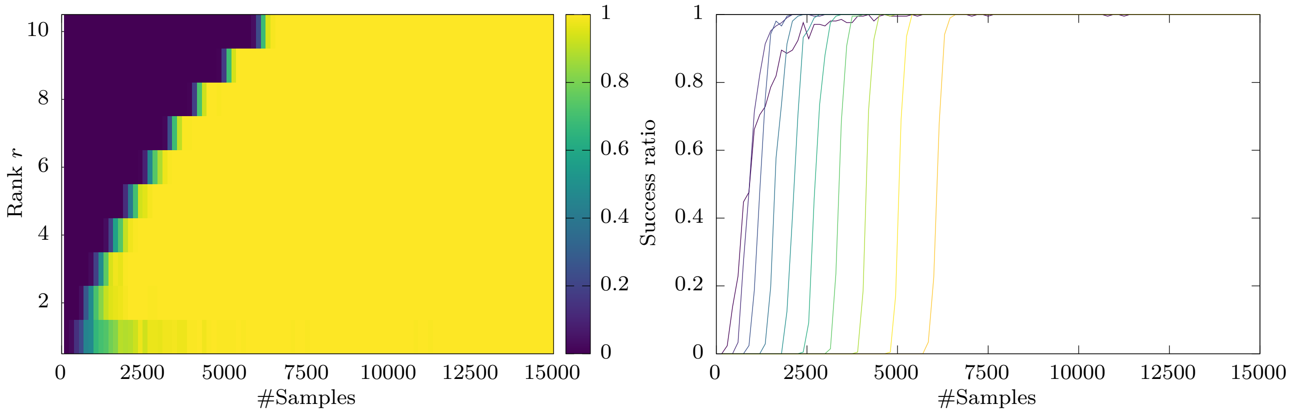


(c) Average relative residual of the solution found by the algorithm.

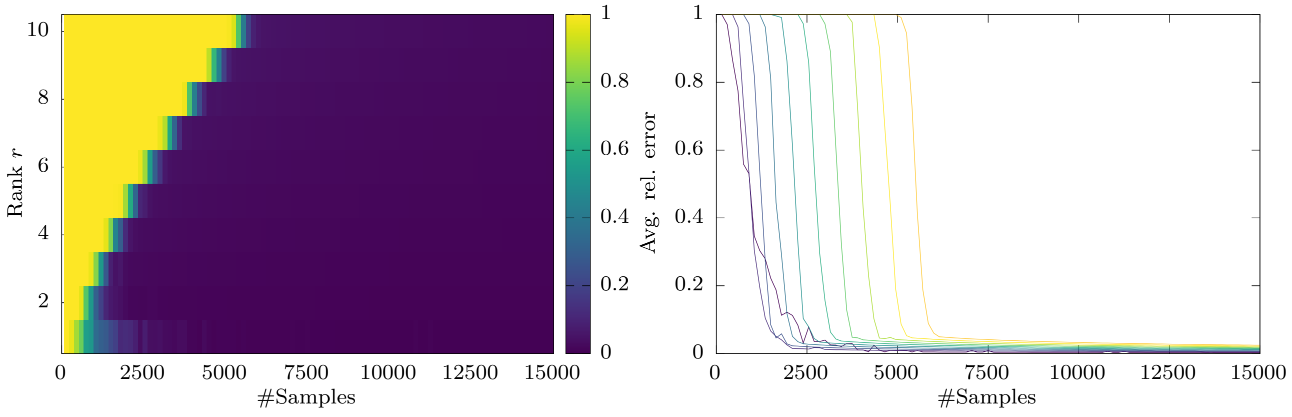


(d) Average number of iterations performed by the algorithm.

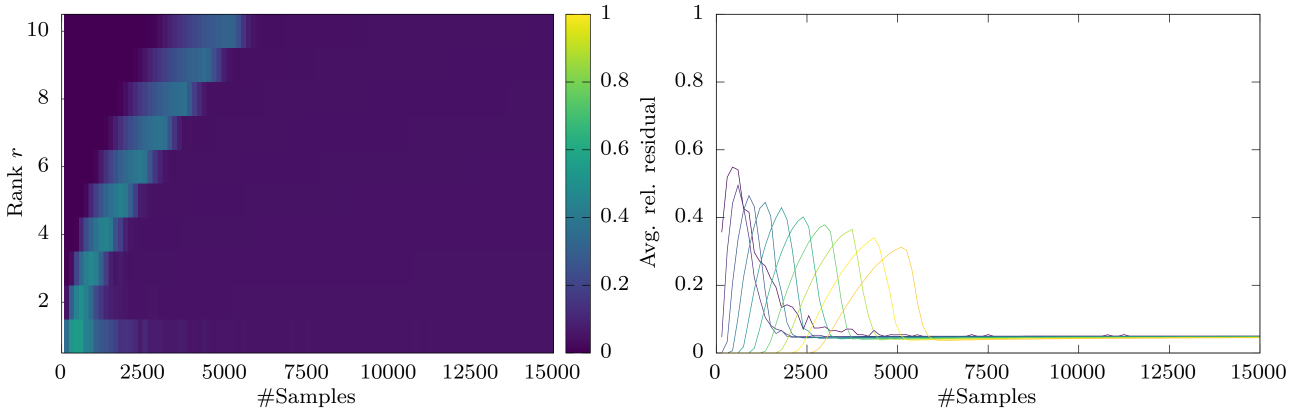
Figure B.2: Numerical results for low rank tensor recovery using the ASD algorithm with single entries of the tensor (completion setting). The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and varying uniform rank r.



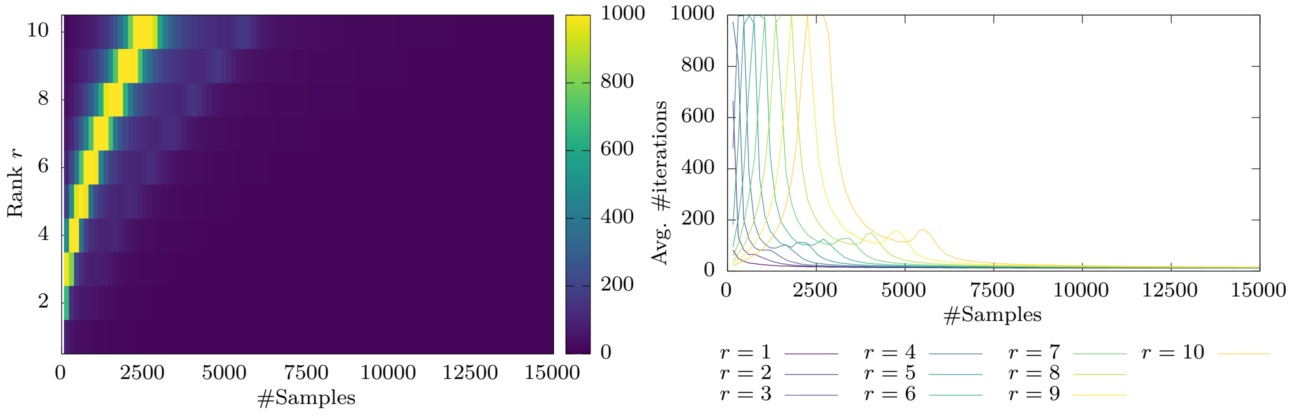
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

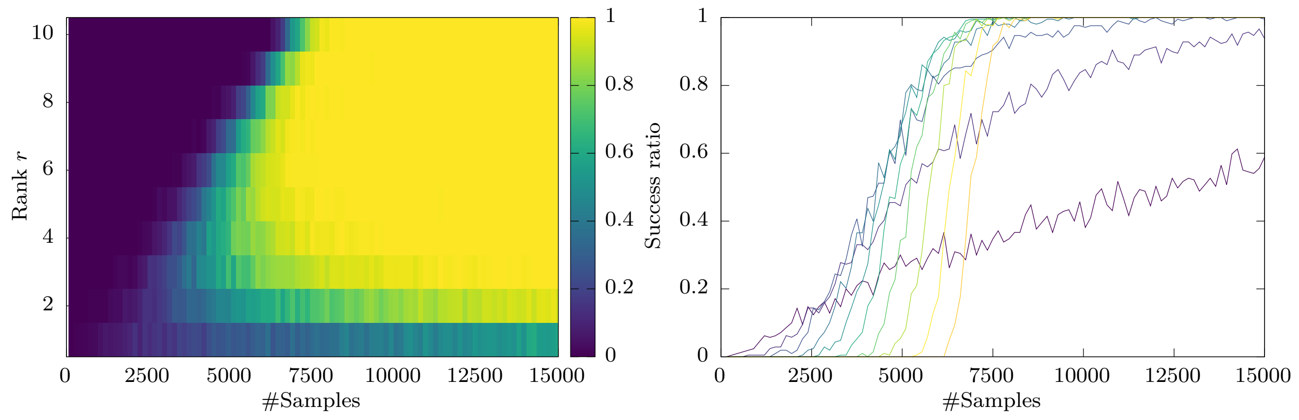


(c) Average relative residual of the solution found by the algorithm.

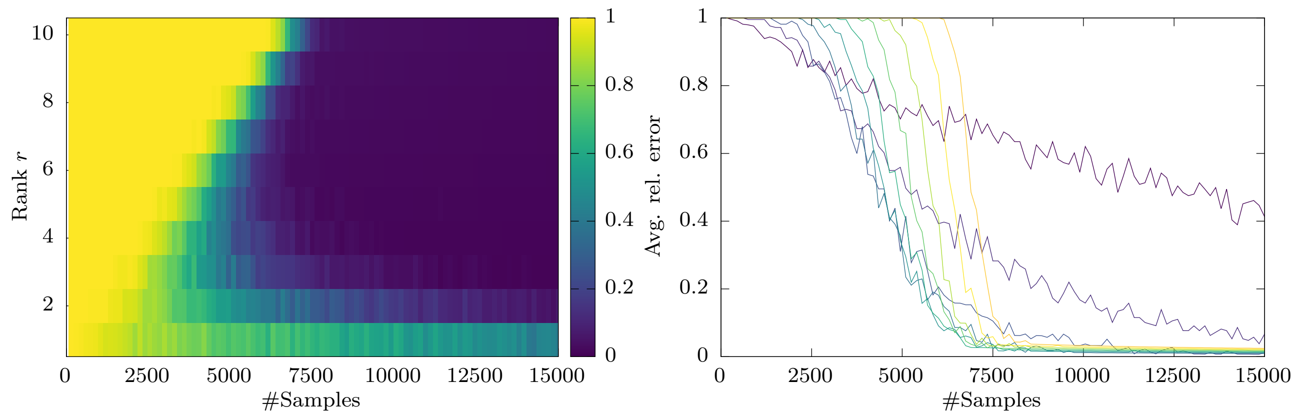


(d) Average number of iterations performed by the algorithm.

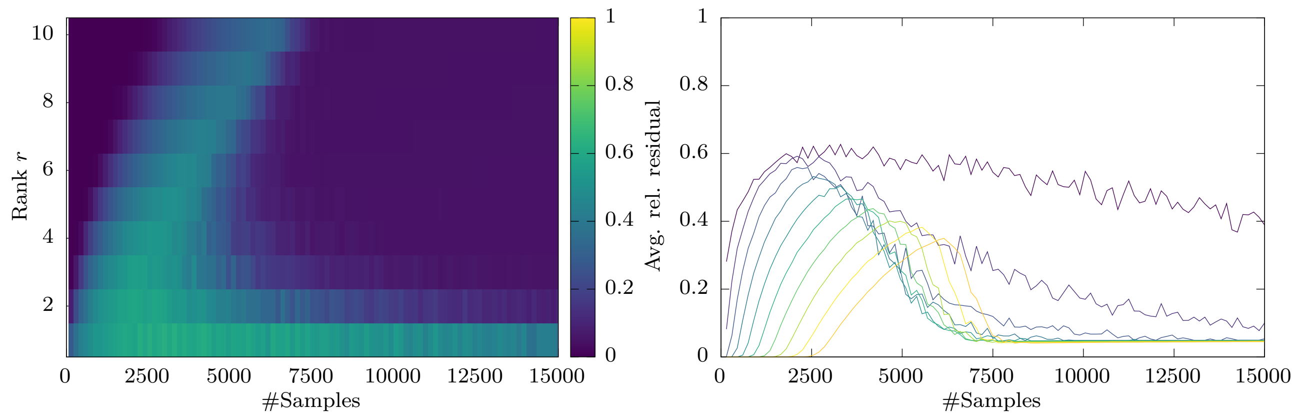
Figure B.3: Numerical results for low rank tensor recovery using the ASD algorithm with noisy rank one samples. The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and varying uniform rank r. The noise amplitude is $\tau = 0.05$.



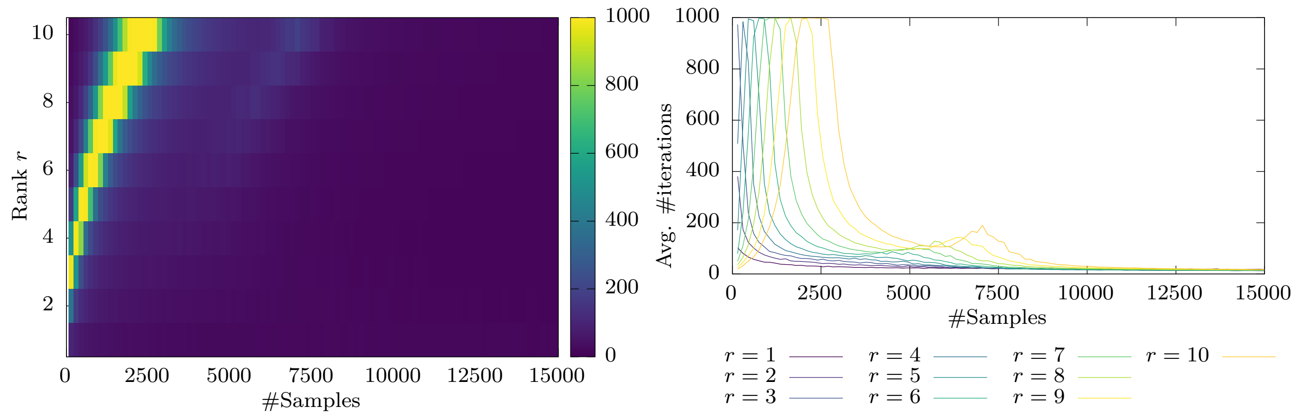
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

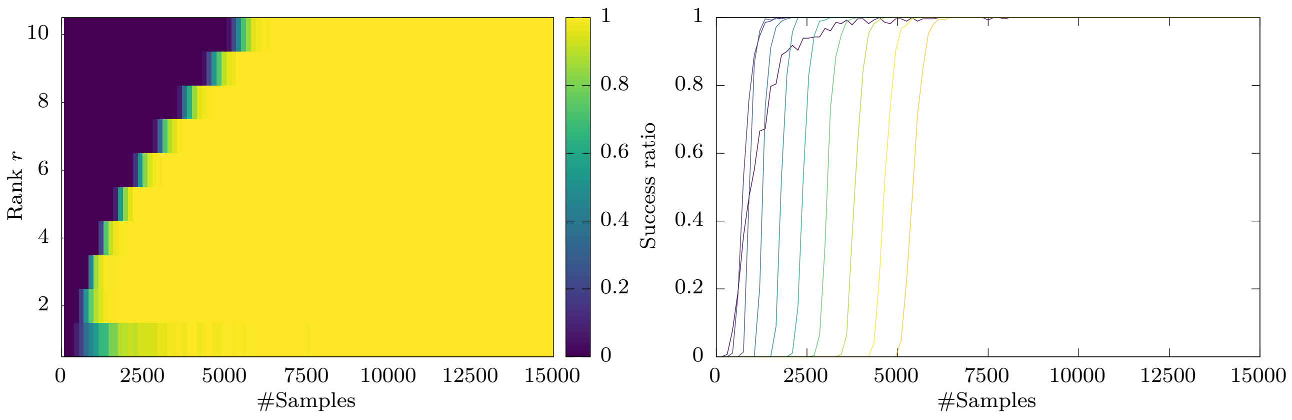


(c) Average relative residual of the solution found by the algorithm.

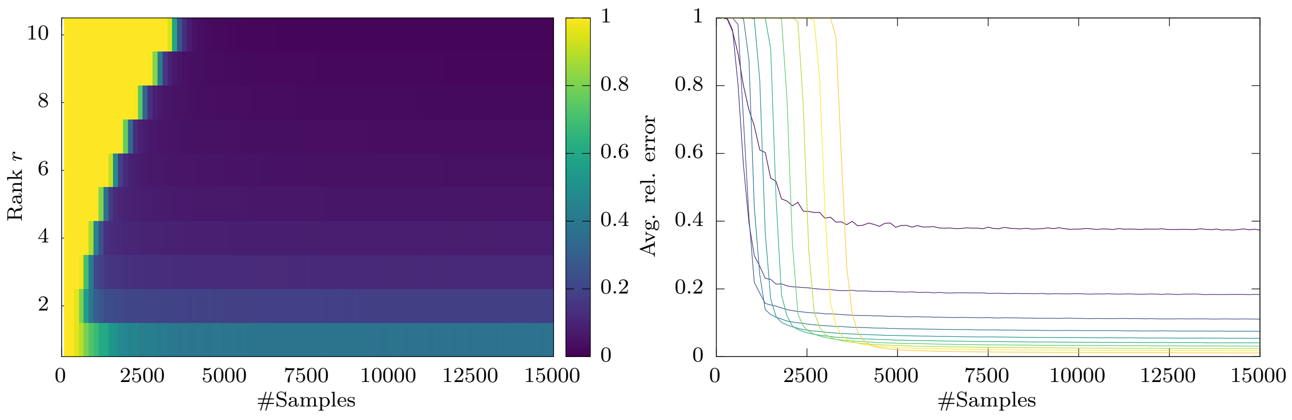


(d) Average number of iterations performed by the algorithm.

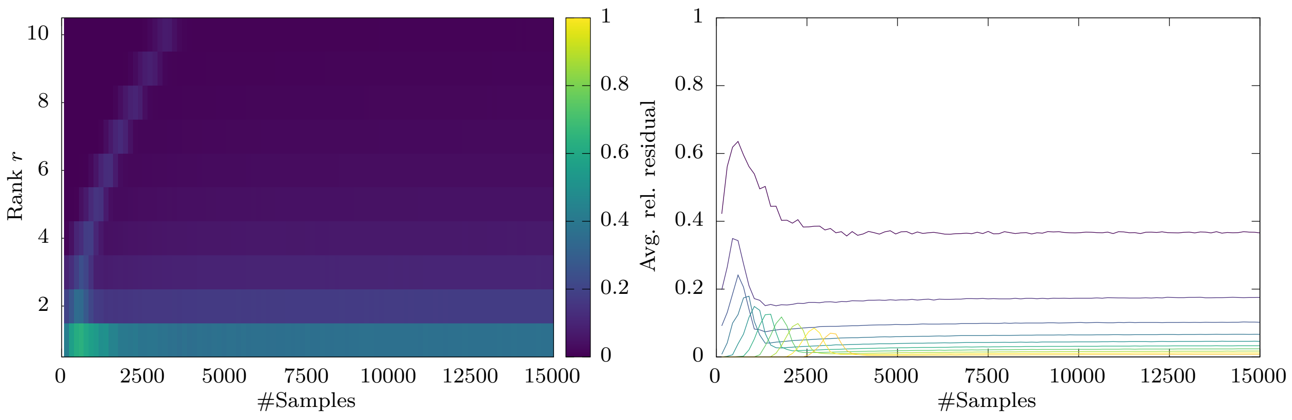
Figure B.4: Numerical results for low rank tensor recovery using the ASD algorithm with noisy measurements of single entries (completion setting). The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and varying uniform rank r. The noise amplitude is $\tau = 0.05$.



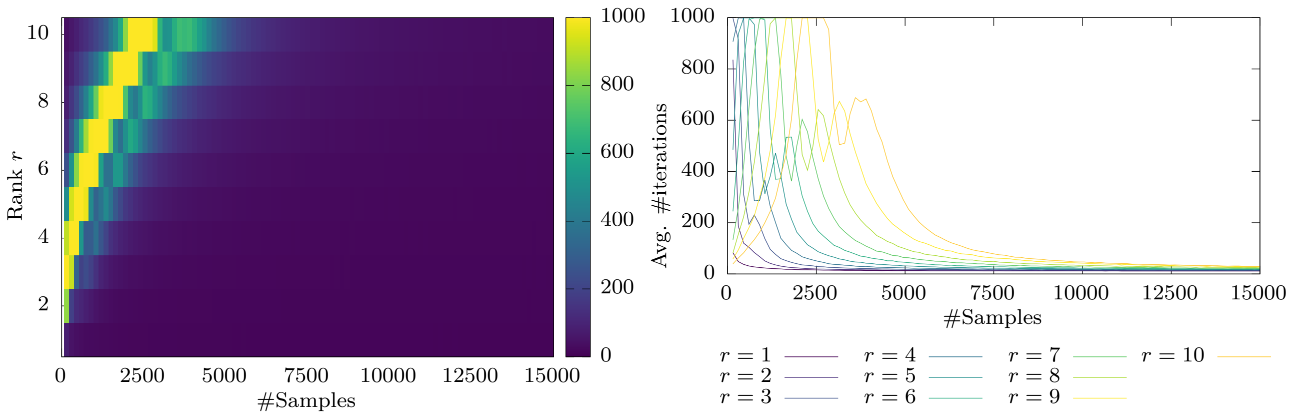
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

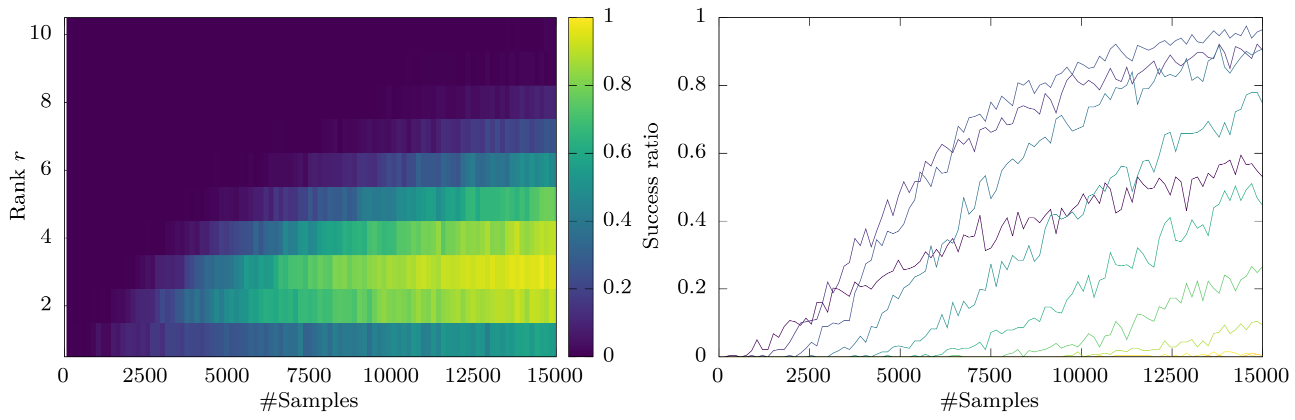


(c) Average relative residual of the solution found by the algorithm.

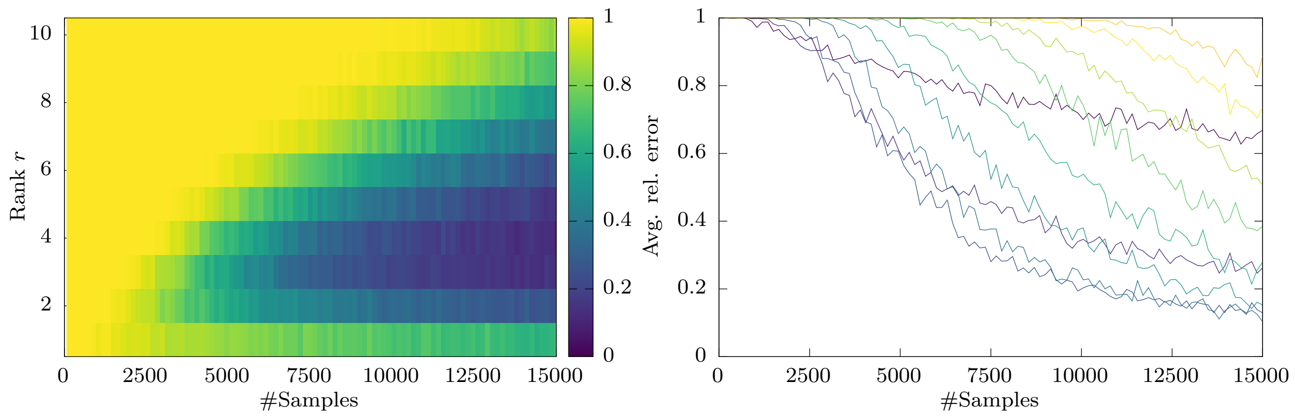


(d) Average number of iterations performed by the algorithm.

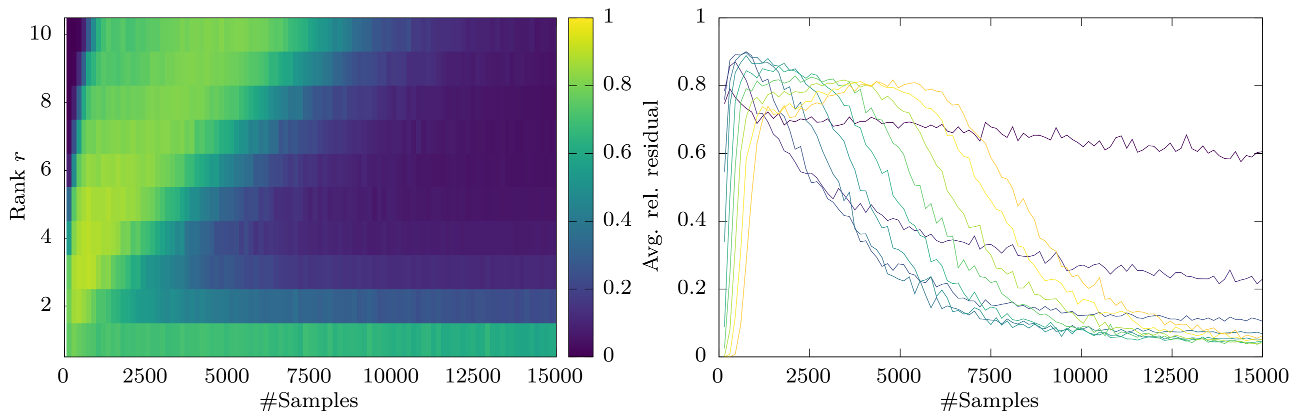
Figure B.5: Numerical results for low rank tensor recovery using the ASD algorithm with rank one samples. The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and have approximately quadratically decreasing singular values.



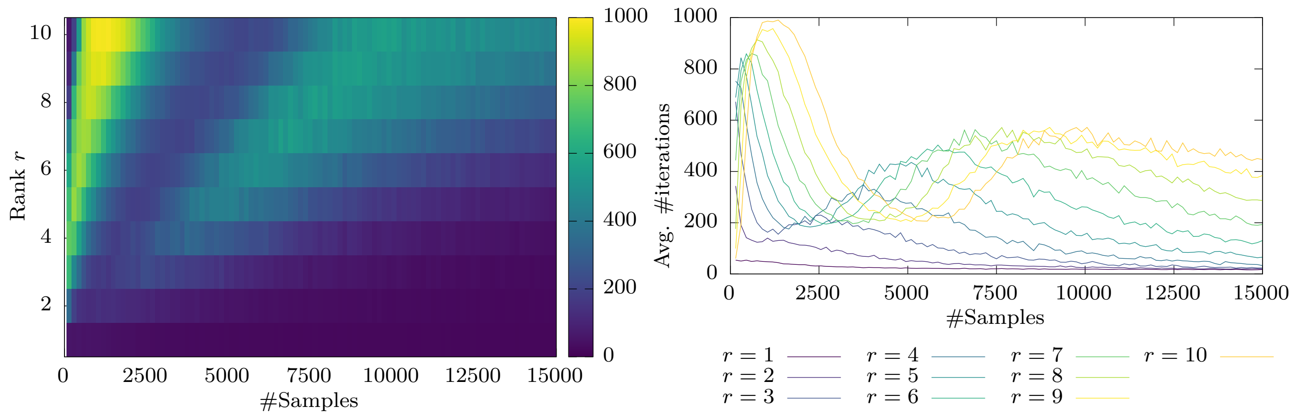
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

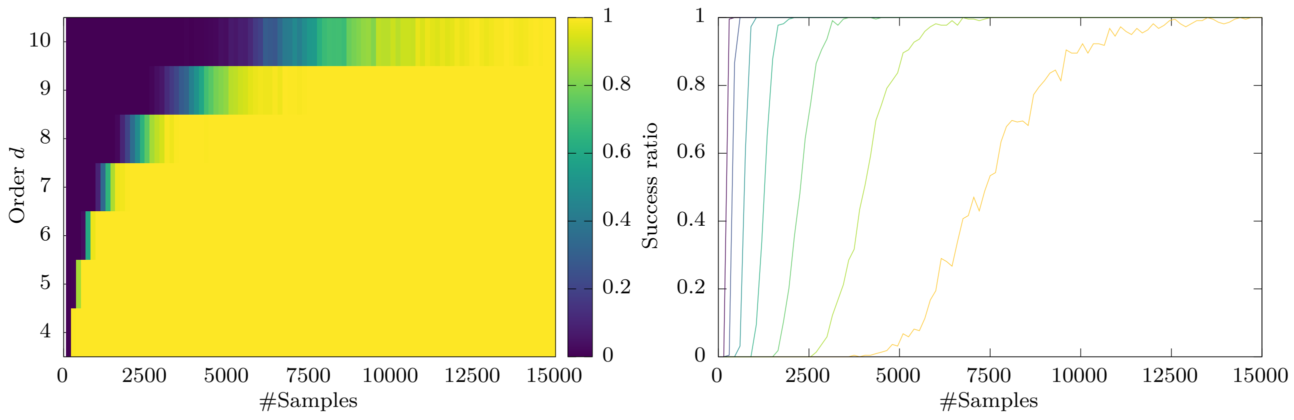


(c) Average relative residual of the solution found by the algorithm.

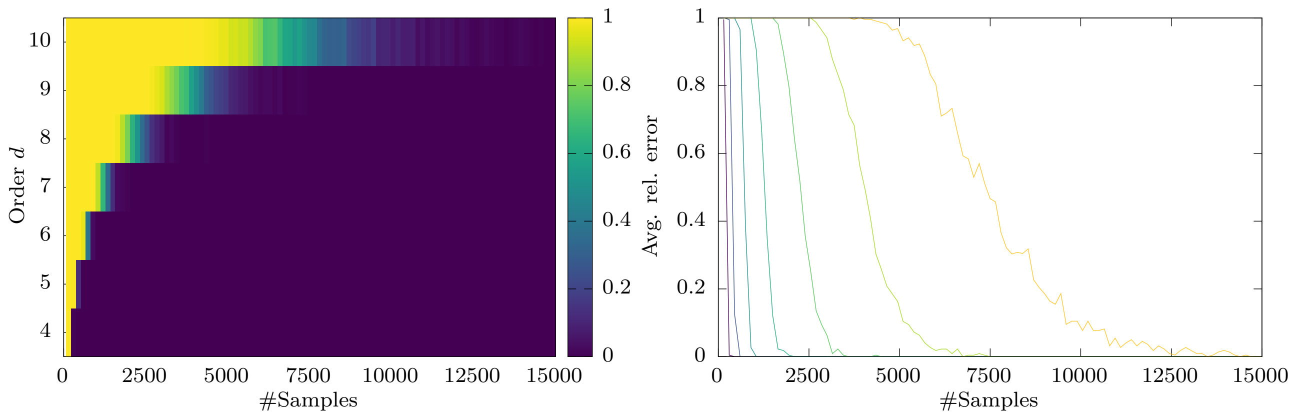


(d) Average number of iterations performed by the algorithm.

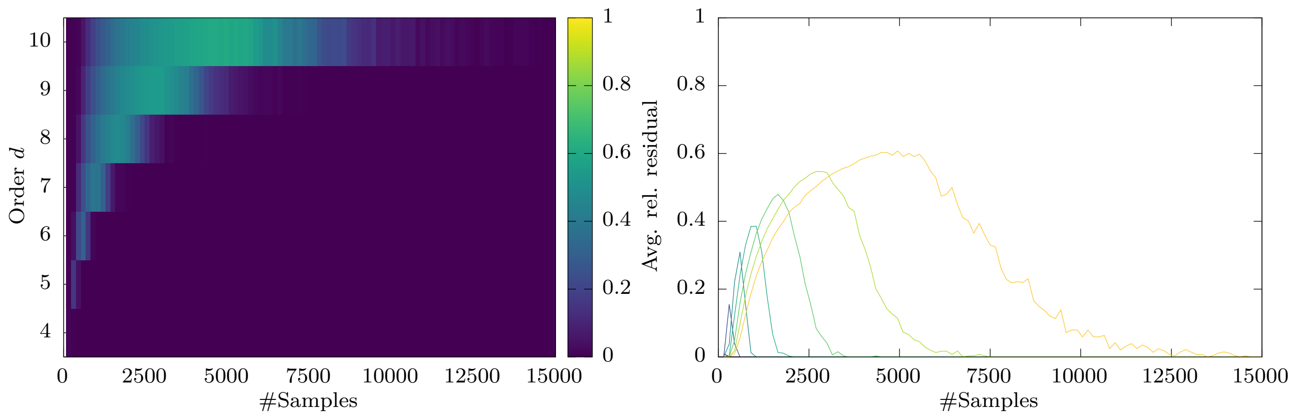
Figure B.6: Numerical results for low rank tensor recovery using the ASD algorithm with single entries of the tensor (completion setting). The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and have approximately quadratically decreasing singular values.



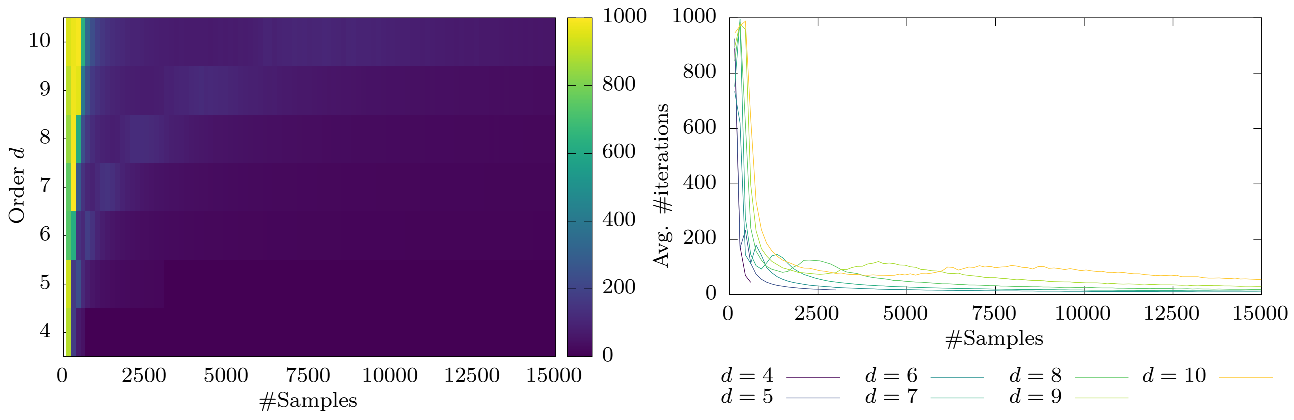
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

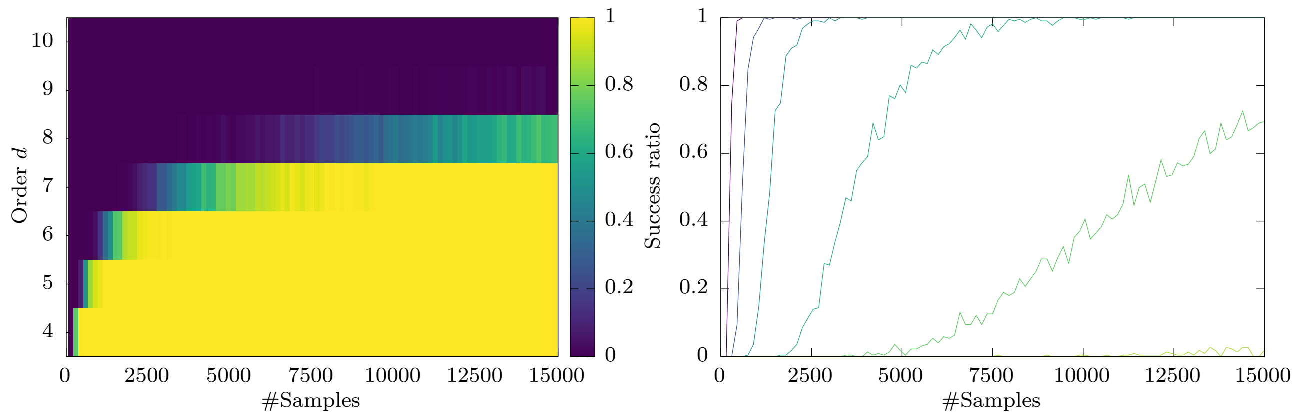


(c) Average relative residual of the solution found by the algorithm.

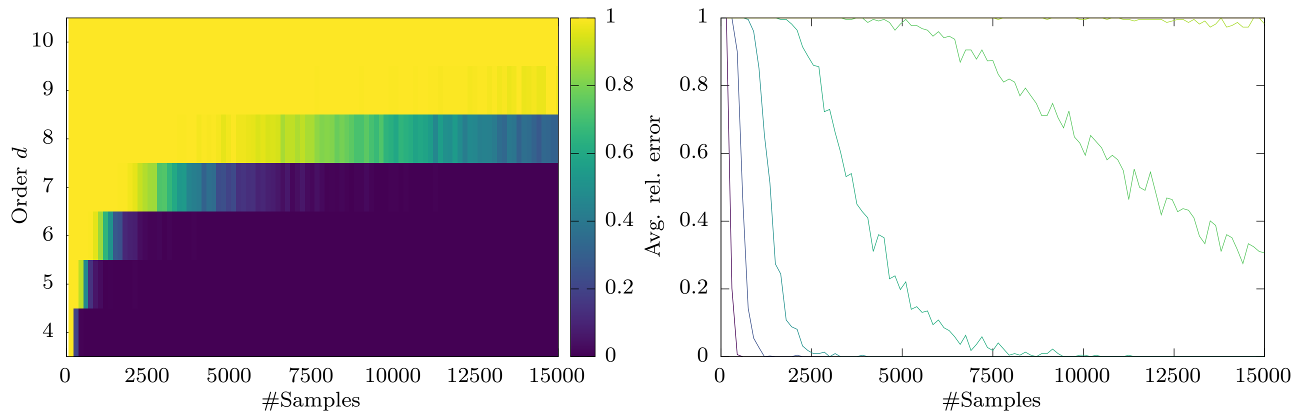


(d) Average number of iterations performed by the algorithm.

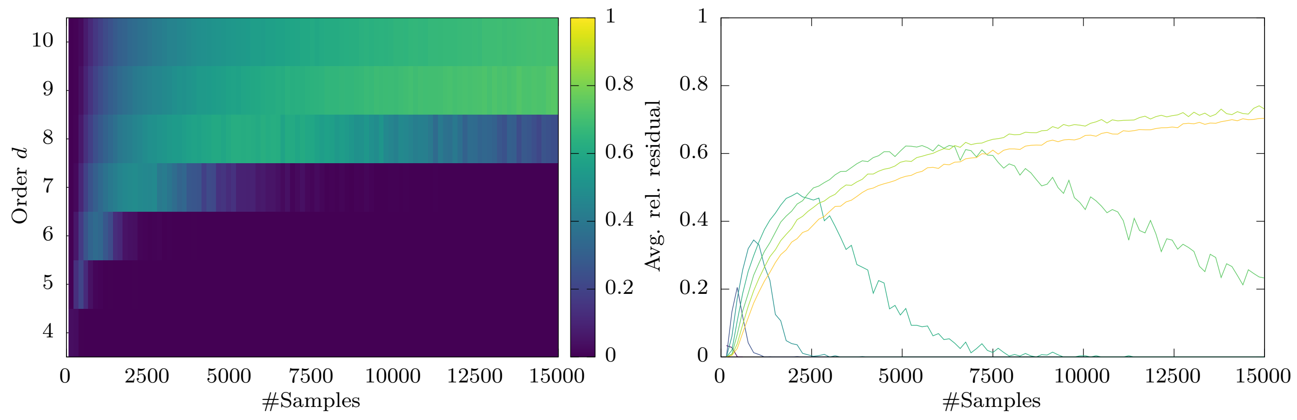
Figure B.7: Numerical results for low rank tensor recovery using the ASD algorithm with rank one samples. The target tensors have uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$, rank $r = 4$ and varying order d.



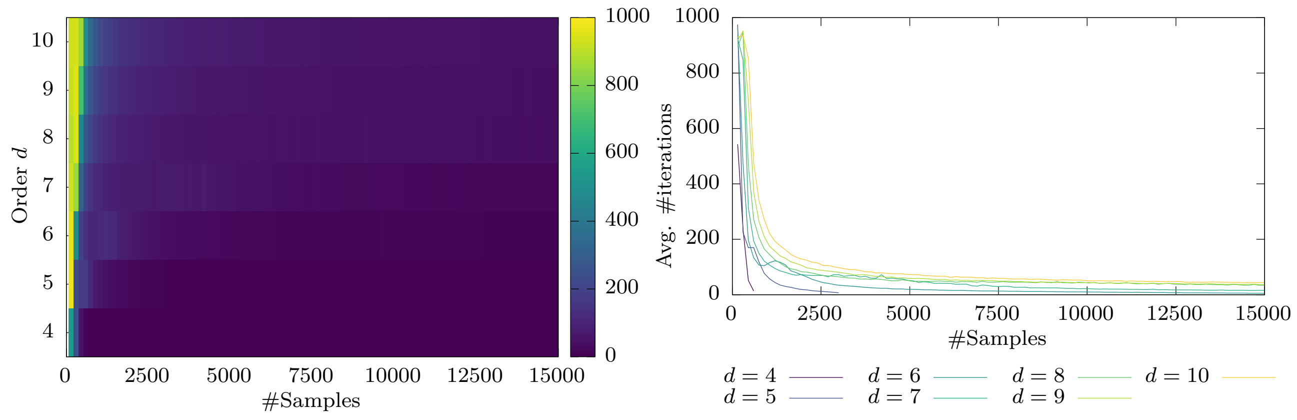
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

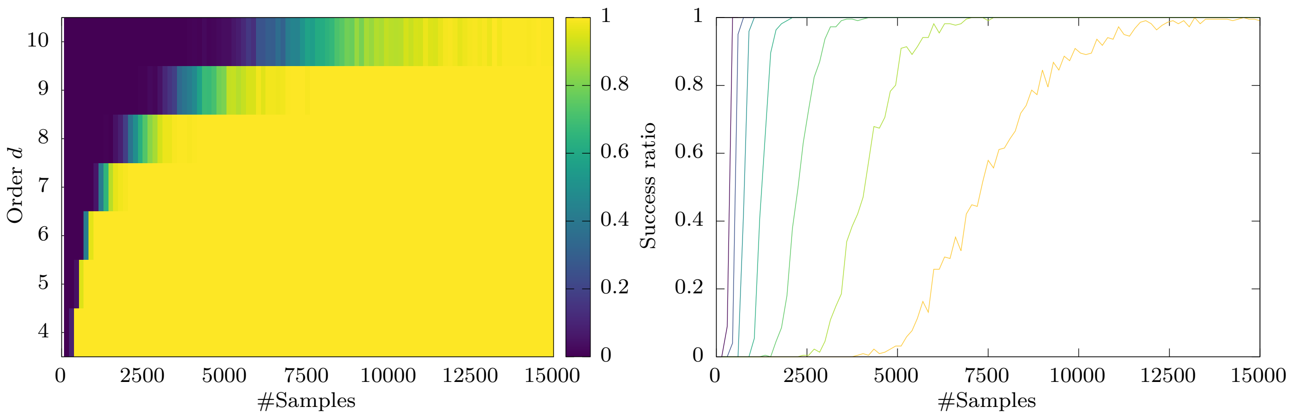


(c) Average relative residual of the solution found by the algorithm.

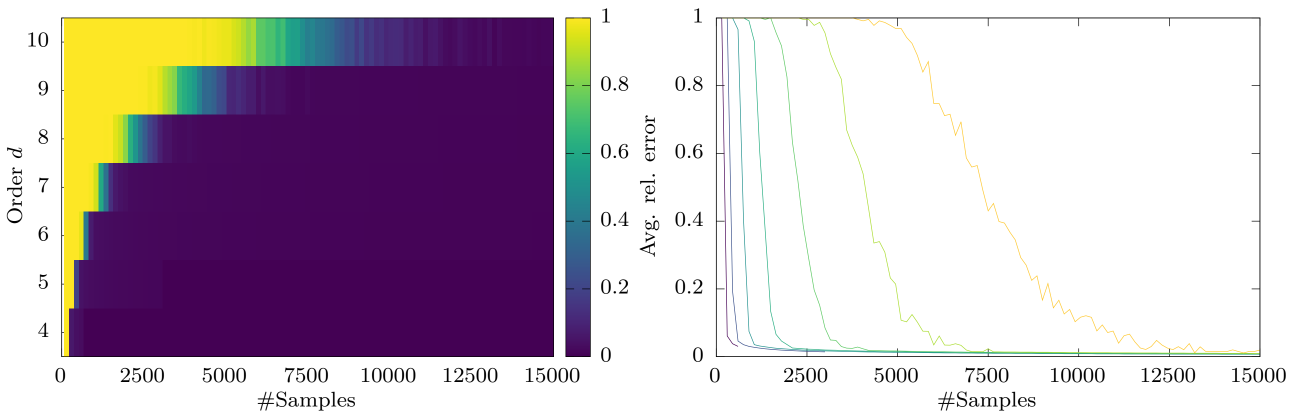


(d) Average number of iterations performed by the algorithm.

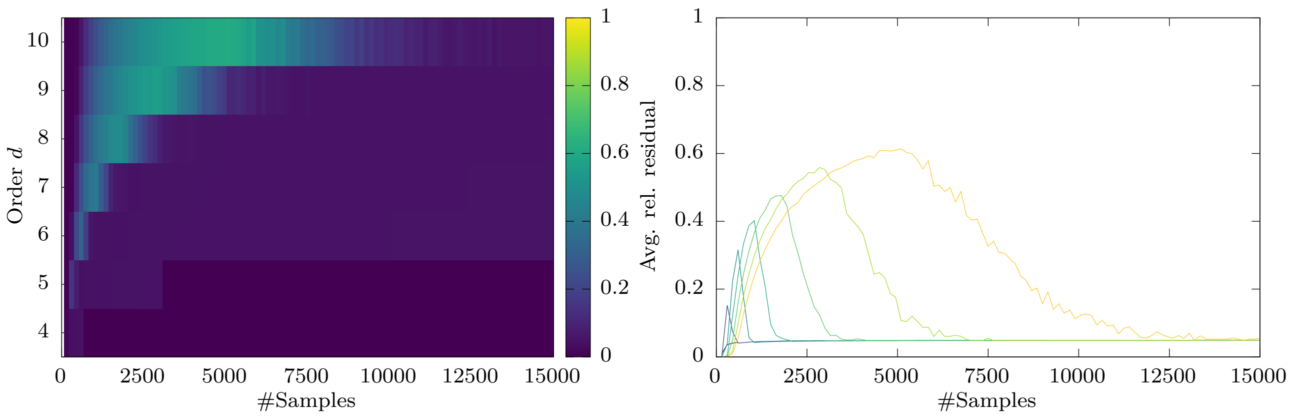
Figure B.8: Numerical results for low rank tensor recovery using the ASD algorithm with single entries of the tensor (completion setting). The target tensors have uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$, rank $r = 4$ and varying order d.



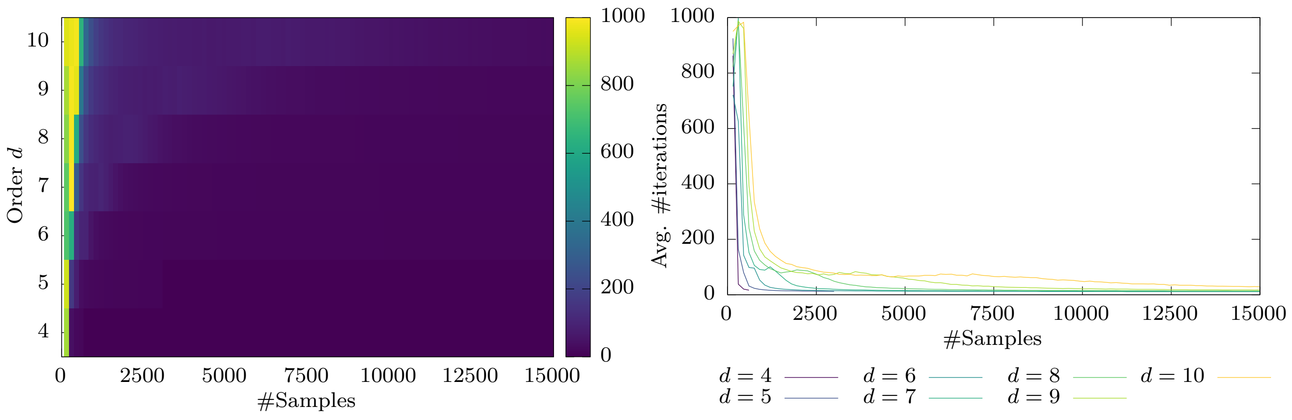
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

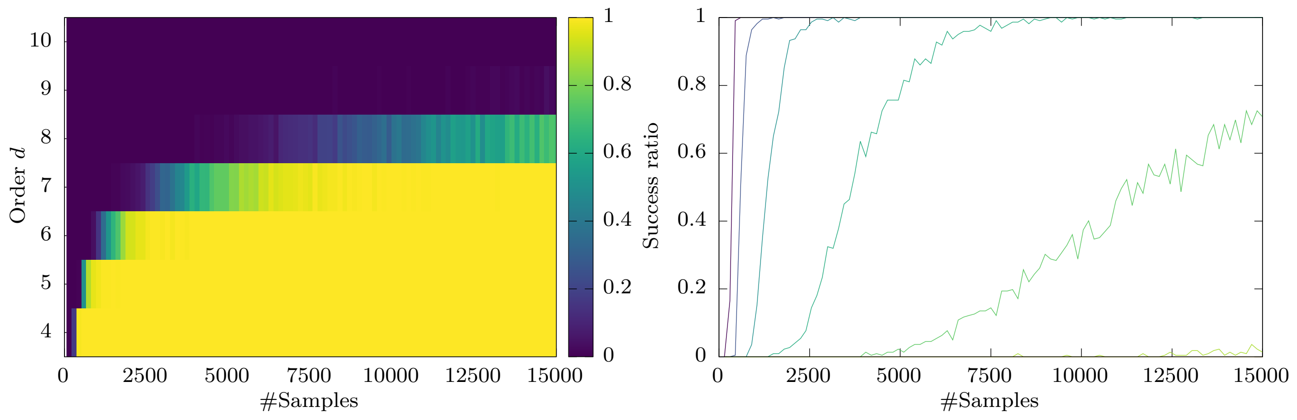


(c) Average relative residual of the solution found by the algorithm.

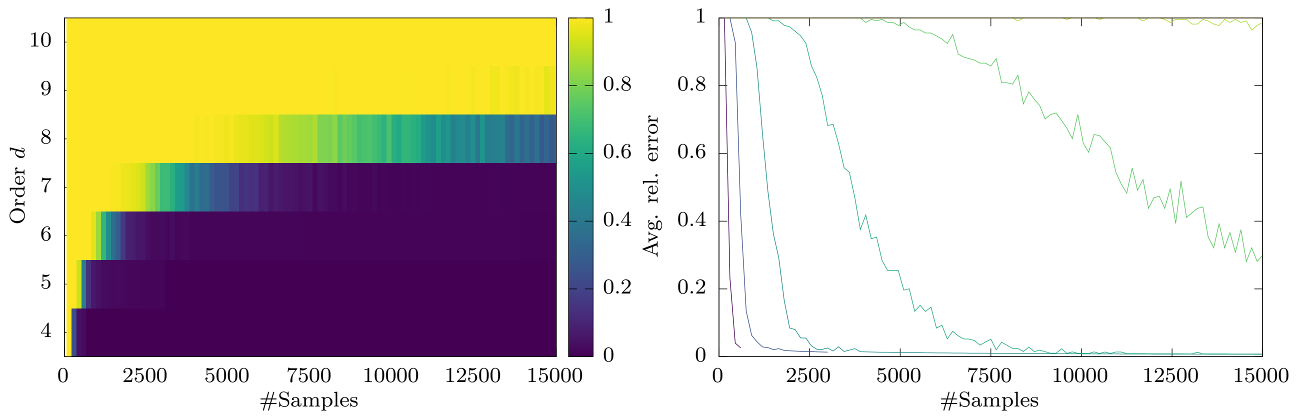


(d) Average number of iterations performed by the algorithm.

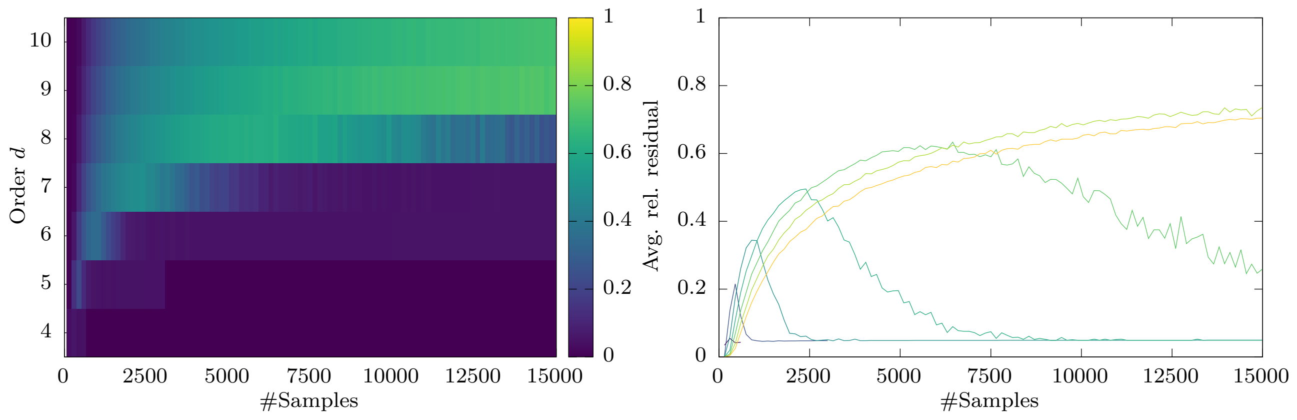
Figure B.9: Numerical results for low rank tensor recovery using the ASD algorithm with noisy rank one samples. The target tensors have uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$, rank $r = 4$ and varying order d. There noise amplitude is $\tau = 0.05$.



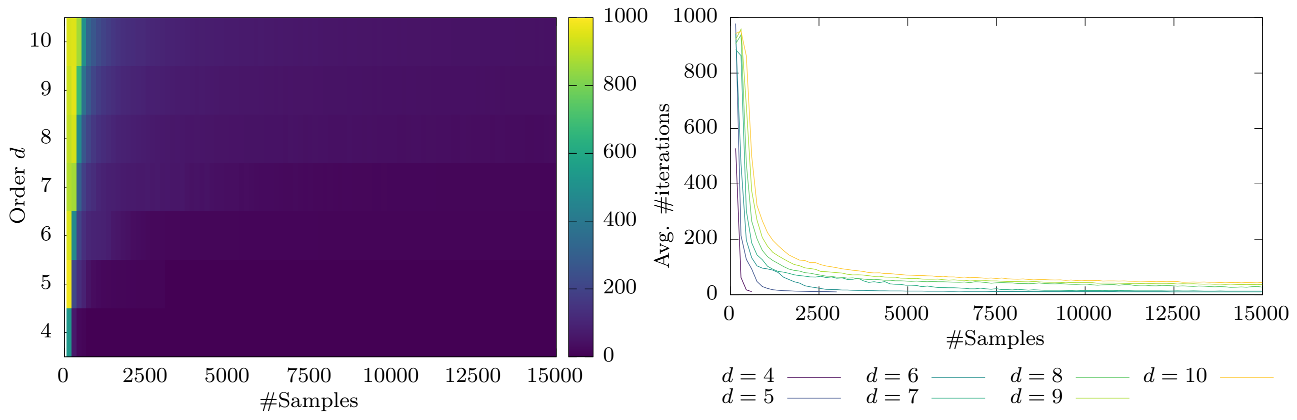
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

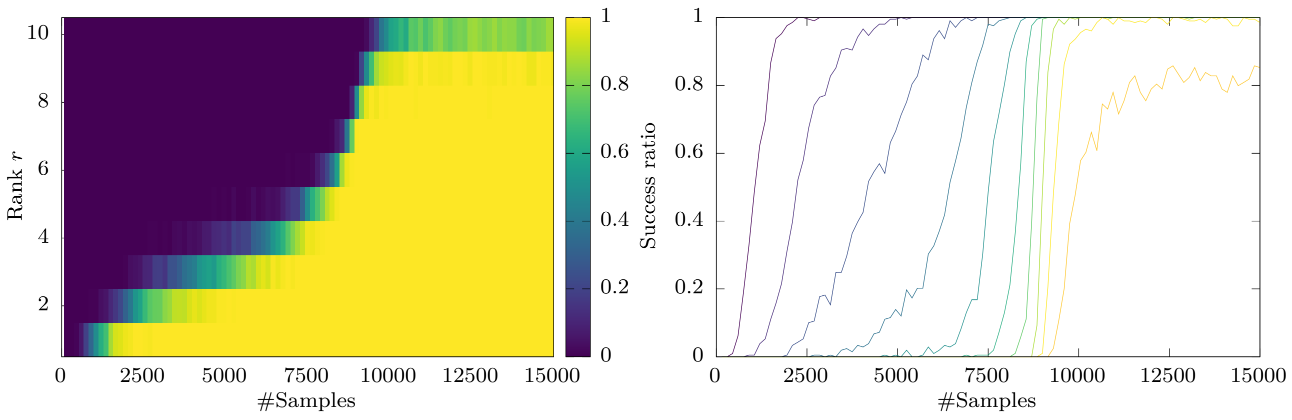


(c) Average relative residual of the solution found by the algorithm.

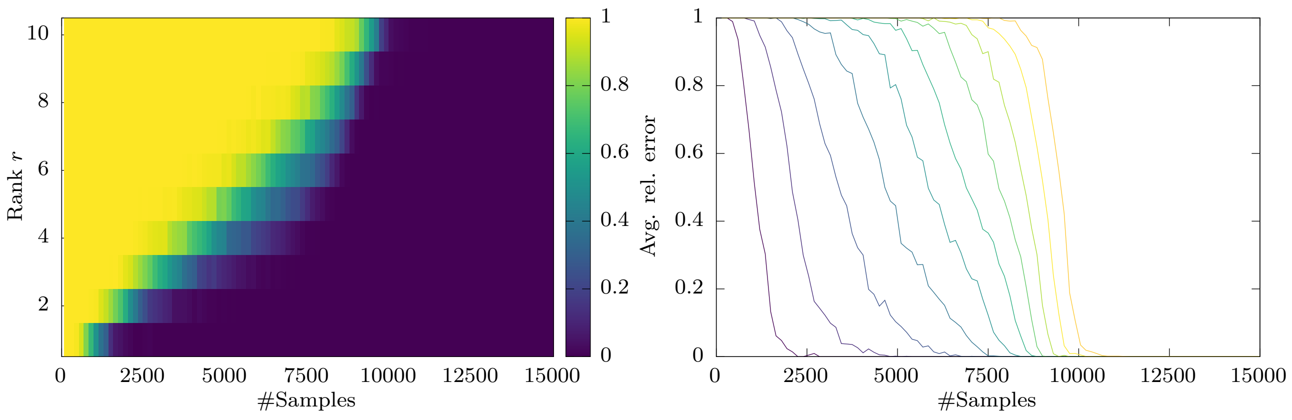


(d) Average number of iterations performed by the algorithm.

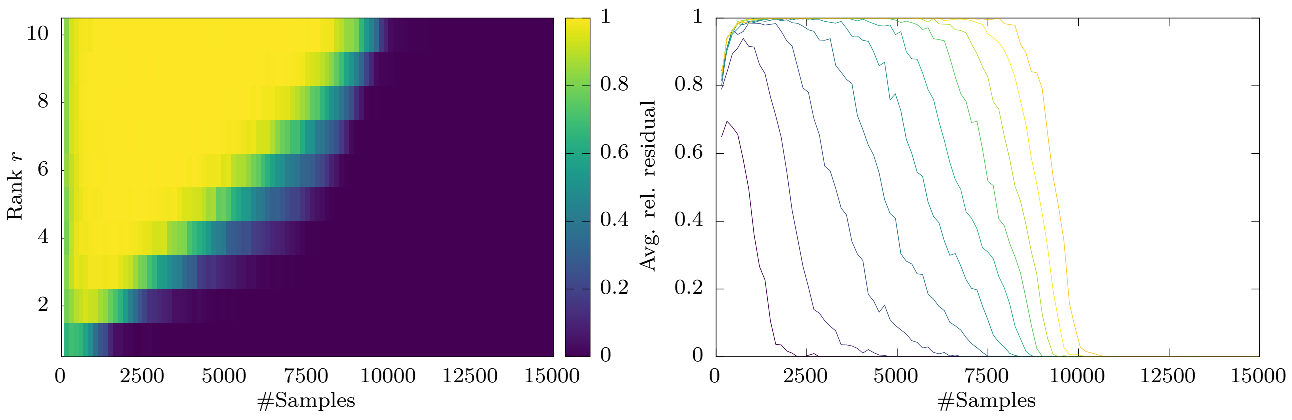
Figure B.10: Numerical results for low rank tensor recovery using the ASD algorithm with noisy measurements of single entries (completion setting). The target tensors have uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$, rank $r = 4$ and varying order d. There noise amplitude is $\tau = 0.05$.



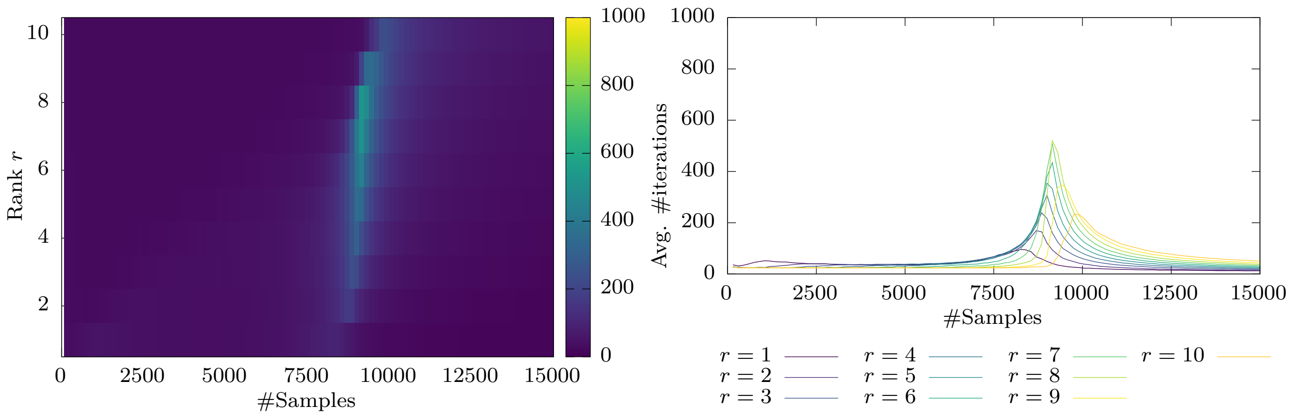
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.

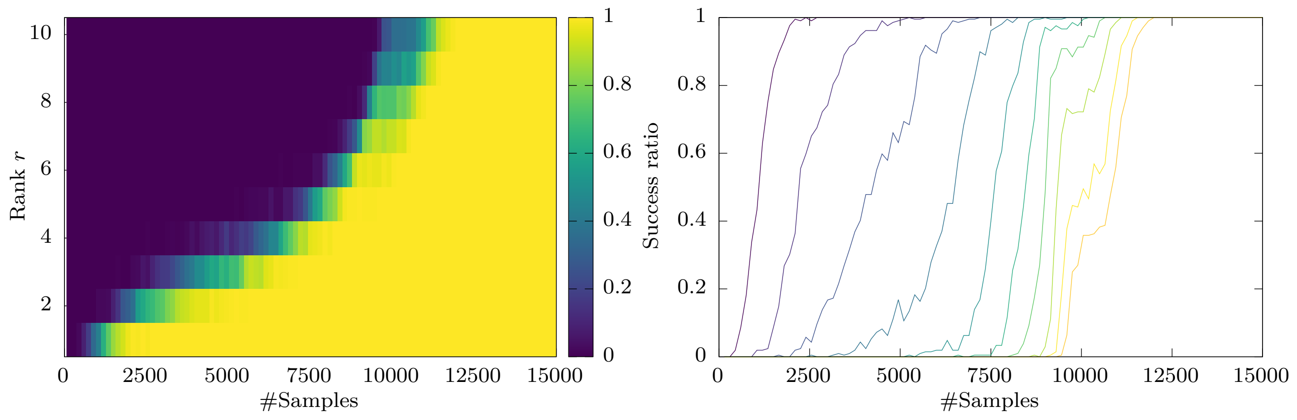


(c) Average relative residual of the solution found by the algorithm.

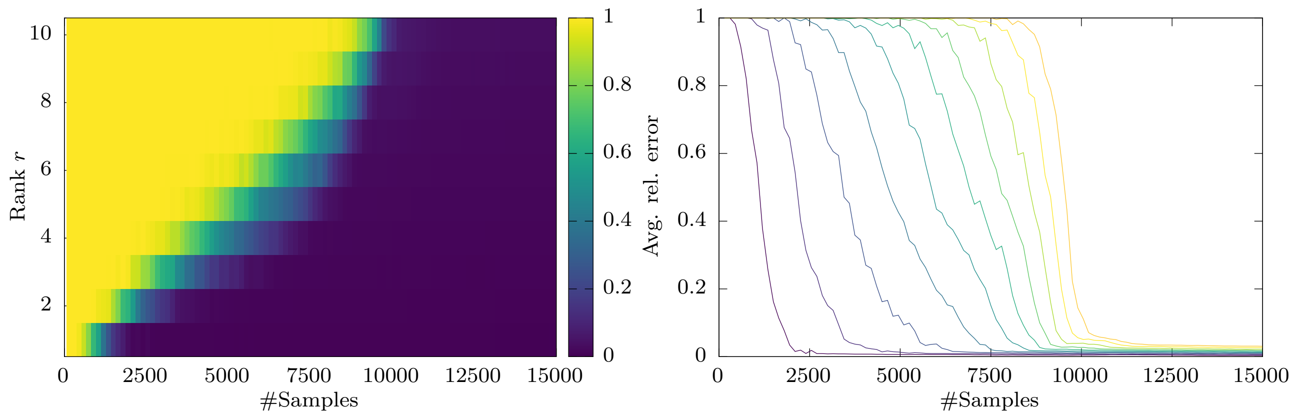


(d) Average number of iterations performed by the algorithm.

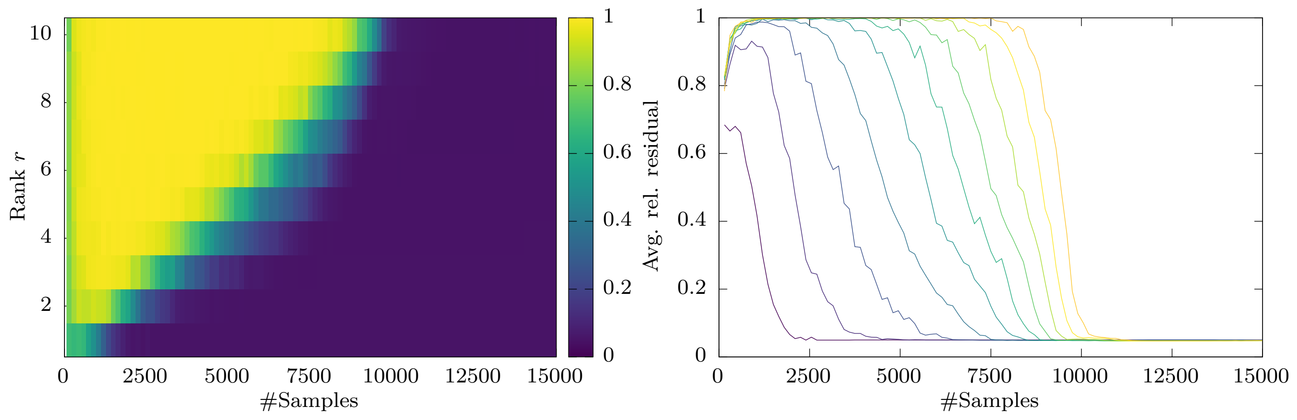
Figure B.11: Numerical results for low rank tensor recovery using the block ASD algorithm with rank one samples. The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and varying uniform rank r.



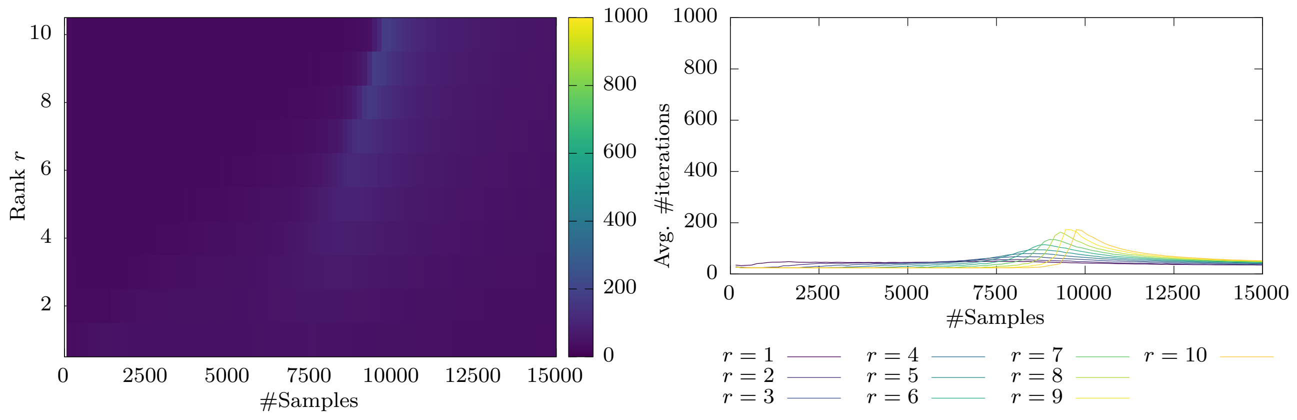
(a) Ratio of successful reconstructions.



(b) Average relative reconstruction error.



(c) Average relative residual of the solution found by the algorithm.



(d) Average number of iterations performed by the algorithm.

Figure B.12: Numerical results for low rank tensor recovery using the block ASD algorithm with noisy rank one samples. The target tensors are of order $d = 5$, uniform local dimensions $n_1 = n_2 = \ldots = n_d = 10$ and varying uniform rank r. The noise amplitude is $\tau = 0.05$.