

KATHOLIEKE UNIVERSITEIT
LEUVEN

Arenberg Doctoral School of Science, Engineering & Technology
Faculty of Engineering
Department of Electrical Engineering (ESAT)

Numerical methods for the best low multilinear rank approximation of higher-order tensors

Mariya Ishteva

Promotors:
Prof. dr. ir. Sabine Van Huffel
Prof. dr. ir. Lieven De Lathauwer

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

December 2009

Numerical methods for the best low multilinear rank approximation of higher-order tensors

Mariya Ishteva

Jury:
Prof. dr. ir. Ann Haegemans, president
Prof. dr. ir. Sabine Van Huffel, promotor
Prof. dr. ir. Lieven De Lathauwer, promotor
Prof. dr. ir. Joos Vandewalle
Prof. dr. ir. Marc Van Barel
Prof. dr. ir. P.-A. Absil (UCL)
Prof. dr. ir. Rodolphe Sepulchre (ULg)
Prof. dr. ir. Lars Eldén (Linköping Univ., Sweden)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

December 2009

© Katholieke Universiteit Leuven – Faculty of Engineering
Kasteelpark Arenberg 1/2200, B-3001 Leuven (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotocopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm or any other means without written permission from the publisher.

D/2009/7515/143

ISBN 978-94-6018-163-4

Foreword

The research work presented in this thesis was carried out at the Department of Electrical Engineering (ESAT) of the Katholieke Universiteit Leuven in the presence of competent and enthusiastic professors and colleagues. This PhD would have been impossible without their professional and moral support.

In particular, first and foremost, I would like to thank my supervisors Prof. Sabine Van Huffel and Prof. Lieven De Lathauwer. Thank you Sabine for believing in me, for your advice, guidance, example and for the nice social events in your house. Lieven, thank you for your competence, for always showing me the right direction, answering my numerous questions, letting me represent you at important conferences en dat je altijd Nederlands hebt gesproken tegen mij.

I am particularly grateful to Prof. P.-A. Absil for agreeing being in my advisory committee, for being an understanding co-author, for his manifold suggestions and improvements and for believing in me. Merci. My thanks go also to Prof. Marc Van Barel for his availability, constructive suggestions, advice and for the careful reading of the thesis. I am honored that Prof. Joos Vandewalle and Prof. Rodolphe Sepulchre agreed to take part in the examination committee and am grateful for their valuable feedback. Thanks to Prof. Ann Haegemans for acting as a chairman and the flexibility in solving administrative issues. Special thanks to Prof. Lars Eldén, not only for coming from abroad for my preliminary defence and for the critical reading of the thesis but also for the invitation to visit the Linköping University in the autumn of 2007. This is greatly appreciated.

There are some more people that I would like to thank personally: Prof. Petko Petkov who introduced me to Sabine; Diana Sima for her helpful advice on both scientific and personal level; Berkant Savas for the nice stay in Linköping and Bart Vandereycken and Michel Journée for the interesting conversations and updates at the DYSCO meetings. Thanks also to all people I met at the conferences for their interest, time and useful discussions and thanks to all the others I forgot to mention in this list.

My current and former colleagues from SCD and in particular BioMed, including our temporary visitors from other universities, provided a wonderful working atmosphere. This is especially true for my office mates through the years, the frequent “guests” from other offices and the everyday company for lunch. The list is too long but I need to mention at least Maria Isabel and Anca. Thanks for all meaningful and meaningless discussions that lead to a continuous good mood. I would also like to thank our secretaries and technical staff for their constant professional help.

Special thanks and warm greetings to all my friends from close and far away, especially to Aylin and Ilia for their long-term friendship and to Dylan for his language corrections, advice on how to do a PhD but mostly for the late evenings’ conversations and for being there.

И накрая, но съвсем не на последно място, специани благодарности на Нина (мама), Камен (тате), Сашо (брат ми) и Никола (половинката) за непрестанната подкрепа, вяра, любов, внимание и насърчения от близо и далеч, както и за много други неща, за които не върви да пиша тук, но ще ви предам лично. Поздрави и целувки ;)

Abstract

In *multilinear algebra*, the basic quantities are generalizations of vectors and matrices, called higher-order tensors. They are used in many application fields, such as higher-order statistics, signal processing and scientific computing. Efficient and reliable algorithms for manipulating these structures are thus highly appreciated.

Matrices are second-order tensors with well-studied properties. The matrix rank is a well-understood concept. In particular, the low-rank approximation of a matrix is essential for various results and algorithms. The solution to the low-rank approximation problem is known and given by the truncated singular value decomposition (SVD). However, the matrix rank and its properties are not easily or uniquely generalizable to higher-order tensors.

This thesis is devoted to a generalization of the matrix column and row rank, namely the *multilinear rank*. We focus on *the best low multilinear rank approximation* of higher-order tensors. Given a higher-order tensor, we are looking for another tensor, as close as possible to the original one and with multilinear rank bounded by prespecified numbers. This approximation is used for dimensionality reduction and signal subspace estimation. Higher-order generalizations of SVD exist but their truncation results in a suboptimal solution of the problem. A refinement by iterative algorithms is required. The higher-order orthogonal iteration is one such algorithm with linear convergence speed.

We aim for conceptually faster algorithms. However, standard optimization algorithms face a difficulty caused by unwanted symmetry property of the cost function. Namely, there are infinitely many equivalent solutions whereas numerical algorithms have nice convergence properties if the solutions are isolated. We remove the symmetry problem by working on *quotient matrix manifolds*, a concept studied in the field of *optimization on manifolds*. We develop three new algorithms, based on Newton's method, trust-region scheme and conjugate gradients. We also discuss the issue of local minima and consider a particular application of the algorithms.

Notation

List of symbols

$a, b, \ldots, \alpha, \beta, \ldots$	scalars
$\mathbf{a}, \mathbf{b}, \ldots$	vectors
$\mathbf{A}, \mathbf{B}, \ldots$	matrices
$\mathscr{A}, \mathscr{B}, \ldots$	tensors
$a_{ijk}, (\mathscr{A})_{ijk}$	element of $\mathscr{A}$ at position (i, j, k)
$\mathscr{A}_{i=\alpha}$	subtensor of $\mathscr{A}$ obtained by fixing the index i
$\mathbf{A}_{(n)}$	mode-n matrix representation of a tensor $\mathscr{A}$
$\mathbf{I}$	identity matrix
$\mathscr{I}$	unit tensor (diagonal tensor with ones on its diagonal)
$\mathbf{H}$	Hankel matrix
$\mathbf{U}, \mathbf{V}, \mathbf{W}$	singular vector matrices
$I, I_1, I_2, I_3, N, R, \ldots$	special scalars and upper bounds
$exp\{\cdot\}$	exponential function
j	$\sqrt{-1}$
π	[1] $\approx 3.141592653589793$ or [2] the quotient map
$\hat{\mathscr{A}}$	low multilinear rank approximation of $\mathscr{A}$
$\hat{x}$	estimate of x
σ_i	i^{th} singular value
$\sigma_i^{(n)}$	i^{th} mode-n singular value
grad	gradient
Hess	Hessian
M	manifold
ξ, η	update/tangent vectors
$\mathscr{V}_{\mathbf{X}}$	vertical space at $\mathbf{X}$
$\mathscr{H}_{\mathbf{X}}$	horizontal space at $\mathbf{X}$
$\overline{\xi}$	horizontal lift of ξ
$P^h_{\mathbf{X}}$	orthogonal projection onto the horizontal space at $\mathbf{X}$

List of sets

$\mathbb{R}$	the set of real numbers
$\mathbb{C}$	the set of complex numbers
$\mathbb{R}^{m\times n}$	real $(m\times n)$-matrix
$\mathbb{R}^{I_1\times I_2\times I_3}$	real $I_1\times I_2\times I_3$ tensor
$\mathbb{R}_*^{n\times p}$	the set of all full-rank $(n\times p)$-matrices, $n\geq p$, $\mathbb{R}_*^{n\times p}=\{\mathbf{X}\in\mathbb{R}^{n\times p}:\det(\mathbf{X}^T\mathbf{X})\neq 0, n\geq p\}$
O_p	the orthogonal group (the set of all orthogonal $(p\times p)$-matrices), $O_p=\{\mathbf{X}\in\mathbb{R}^{p\times p}:\mathbf{X}^T\mathbf{X}=\mathbf{X}\mathbf{X}^T=\mathbf{I}\}$
$St(p,n)$	the Stiefel manifold (the set of all column-wise orthonormal $(n\times p)$-matrices), $St(p,n)=\{\mathbf{X}\in\mathbb{R}^{n\times p}:\mathbf{X}^T\mathbf{X}=\mathbf{I}, n\geq p\}$
$\mathscr{S}_{\text{sym}}(p)$	the set of all symmetric matrices of order p, $\mathscr{S}_{\text{sym}}(p)=\{\mathbf{S}\in\mathbb{R}^{p\times p}:\mathbf{S}^T=\mathbf{S}\}$
$\mathscr{S}_{\text{skew}}(p)$	the set of all skew-symmetric (or antisymmetric) matrices of order p, $\mathscr{S}_{\text{skew}}(p)=\{\mathbf{\Omega}\in\mathbb{R}^{p\times p}:\mathbf{\Omega}^T=-\mathbf{\Omega}\}$

Basic operations

$\mathbf{A}^T$	transpose of the matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	inverse of $\mathbf{A}$
$[\mathbf{a}\,\mathbf{b}]$	matrix with columns $\mathbf{a}$ and $\mathbf{b}$
$\text{vec}(\mathbf{A})$	vector representation of $\mathbf{A}$ (all columns of $\mathbf{A}$ stacked after each other)
$\lvert a\rvert$	absolute value of a
$\langle\cdot,\cdot\rangle$	inner/scalar product
$\lVert\mathbf{A}\rVert$	Frobenius norm, $\sqrt{\text{trace}(\mathbf{A}^T\mathbf{A})}$
$\det(\mathbf{A})$	determinant of $\mathbf{A}$
$\text{diag}(\mathbf{v})$	square diagonal matrix with vector $\mathbf{v}$ as diagonal
$\text{trace}(\mathbf{A})$	trace of the matrix $\mathbf{A}$, $\sum a_{ii}$
$\text{sym}(\mathbf{B})$	$(\mathbf{B}+\mathbf{B}^T)/2$
$\text{skew}(\mathbf{B})$	$(\mathbf{B}-\mathbf{B}^T)/2$
$\text{rank}(\cdot)$	rank
$\text{rank}_n(\mathscr{A})$	mode-n rank of $\mathscr{A}$
$\text{im}(f)$	image of f
$\text{col}(\mathbf{Y})$	the column space of $\mathbf{Y}$, $\{\mathbf{Y}\mathbf{a}:\mathbf{a}\in\mathbb{R}^p\}$
$[\mathbf{Y}]$	the equivalence class of $\mathbf{Y}$
$\mathbf{\Sigma}$	sum
$\prod$	product
$\times$	Cartesian product, $A\times B:=\{(a_i,b_j)\}$
$\otimes$	Kronecker product, $\mathbf{A}\otimes\mathbf{B}:=[a_{ij}\mathbf{B}]$
$\mathbf{a}\circ\mathbf{b}$	outer product of vectors $\mathbf{a}$ and $\mathbf{b}$
$\mathscr{A}\bullet_n\mathbf{M}$	mode-n product of a tensor $\mathscr{A}$ and a matrix $\mathbf{M}$
$a\sim b$	a is equivalent to b
$a\approx b$	a is approximately equal to b
$\in$	belongs to

Abbreviations

ALS	Alternating Least Squares
CANDECOMP	Canonical Decomposition
CG	Conjugate Gradients
EDS model	Exponentially Damped Sinusoidal model
EEG	Electroencephalography, Electroencephalogram
e.g.	for example (*exempli gratia*)
HOSVD	Higher-Order Singular Value Decomposition
HOOI	Higher-Order Orthogonal Iteration
HSVD	Hankel Singular Value Decomposition
HTLS	Hankel Total Least Squares
ICA	Independent Component Analysis
i.e.	that is (*id est*)
LS	Least Squares
MEG	Magneto-Encephalography
NMR	Nuclear Magnetic Resonance
PARAFAC	Parallel Factor decomposition
RRMSE	Relative Root Mean Square Error
SD	Steepest Descent
SVD	Singular Value Decomposition
TLS	Total Least Squares
TR	Trust Region

Contents

Contents ix

List of Figures xiii

Lists of Tables and Algorihtms xvii

1 Introduction 1

1.1 Contributions of the thesis 3

1.2 Chapter-by-chapter overview 4

I Background material 9

2 Multilinear algebra 11

2.1 Basic definitions, operations on and properties of higher-order tensors . . 11

2.1.1 Higher-order tensor 12

2.1.2 Rank and multilinear rank 13

2.1.3 Basic operations 14

2.1.4 Main properties 16

2.2 Best low multilinear rank approximation 17

2.3 Motivation 18

2.4 HOSVD, HOOI and related methods 20

2.4.1 Higher-order singular value decomposition 20
2.4.2 Higher-order orthogonal iteration 23
2.4.3 Other methods in the literature 24
2.5 Conclusions 24

3 Matrix manifolds **25**
3.1 Motivation: Invariance property 26
3.2 General concepts 26
3.3 Quotient matrix manifolds 28
3.4 Grassmann manifold 29
3.5 The manifold $\mathbb{R}_*^{n\times p}/O_p$ 31
3.6 Retraction and vector transport 32
3.6.1 Retraction 32
3.6.2 Vector transport 34
3.7 Conclusions 35

II Algorithms **37**

4 A geometric Newton method for Oja's vector field **39**
4.1 Introduction 39
4.2 Plain Newton method for Oja's vector field 41
4.3 Newton's method on $\mathbb{R}_*^{n\times p}/O_p$ 43
4.4 A geometric Newton method for Oja's vector field 45
4.5 Numerical experiments 47
4.6 Conclusions 51

5 Geometric Newton algorithm **53**
5.1 Reformulation of the problem 53
5.1.1 The new cost function 54

5.1.2 The invariance property 55
5.2 Differential-geometric Newton method for the best rank-(R_1, R_2, R_3) approximation 55
5.3 Numerical Experiments 58
5.4 Conclusions 61

6 Trust-region algorithm 63
6.1 Riemannian trust-region scheme 64
6.1.1 Basic trust-region method 64
6.1.2 Trust-region method on a Riemannian manifold 65
6.2 Riemannian trust-region based rank-(R_1, R_2, R_3) approximation 66
6.2.1 Computation of the gradient 68
6.2.2 Computation of the Hessian 68
6.2.3 Summary of the algorithm 70
6.3 Comparison with HOOI 71
6.3.1 Convergence speed 71
6.3.2 Computational cost 73
6.4 Comparison with Newton-type methods 74
6.5 Conclusions 76

7 Conjugate gradient algorithm 79
7.1 Conjugate gradient methods 79
7.1.1 Linear conjugate gradient method 80
7.1.2 Nonlinear conjugate gradient method 81
7.2 Best rank-(R_1, R_2, R_3) approximation with conjugate gradients 82
7.2.1 Conjugate gradients on a Riemannian manifold 82
7.2.2 Implementation and summary of the algorithm 84
7.3 Numerical properties 85
7.3.1 General performance of CG 85

7.3.2 Computational cost 88
7.3.3 Precision and restarts 89
7.4 Comparison with other algorithms 90
7.4.1 Comparison with HOOI 90
7.4.2 Comparison with the trust-region algorithm 91
7.4.3 Summary 92
7.5 Conclusions 93

III Practical experience 95

8 Local minima 97
8.1 Existence of local minima 97
8.2 Properties of local minima 101
8.3 Using the truncated HOSVD as a starting value 105
8.4 Conclusions 106

9 Application: Harmonic analysis 109
9.1 Introduction 110
9.2 Theoretical background 111
9.3 Simulations 114
9.4 Conclusions 116

IV Closing 117

10 Conclusions and open problems 119
10.1 General conclusions of the thesis 119
10.2 Future work and open problems 120

Bibliography 125

List of Figures

1.1 Visualization of a third-order tensor. 1

2.1 Mode-*n* vectors of a $(6 \times 5 \times 4)$-tensor. 12
2.2 Rank-1 tensor as an outer product of three vectors. 13
2.3 Matrix representations of a tensor. 15
2.4 Dimensionality reduction. 19
2.5 Higher-order singular value decomposition. 21

3.1 Manifold M, tangent plane $T_{\mathbf{X}}M$ and a tangent vector ξ at point $\mathbf{X} \in M$. . 27
3.2 Illustration of a quotient manifold. 29
3.3 Retraction. 33
3.4 Vector transport. 34

4.1 Plain Newton method. 48
4.2 Geometric Newton method. 49
4.3 Distribution of the basin of attraction of three eigenspaces of interest. The horizontal axis shows the largest principal angle between $\mathrm{col}(\mathbf{X}(0))$ and the eigenspace under consideration. The vertical axis gives the fraction of initial conditions that led to convergence to the eigenspace. 50
4.4 Illustration of the superlinear convergence of the geometric Newton method. 51

5.1 Evolution of the original cost function (2.2), the difference in cost function between each iteration and the last iterate and the function F from (5.4). The tensor $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ is as in (5.11), with $R_1 = R_2 = R_3 = 5$ and $\sigma = 0.1$ The starting values are taken from the truncated HOSVD and initial 5 iterations of HOOI are performed. 59

5.2 Evolution of the original cost function (2.2), the difference in cost function between each iteration and the last iterate and the function F from (5.4). The tensor $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ has elements uniformly distributed over the interval $[0,1]$ and $R_1 = R_2 = R_3 = 5$. The starting values are taken from the truncated HOSVD and initial 20 iterations of HOOI are performed. 60

6.1 Evolution of the relative norm of the gradient $\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X})$ and the cost function (2.2). The tensor $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ has elements uniformly distributed over the interval $[0,1]$; $R_1 = R_2 = R_3 = 5$. The starting values are taken from the truncated HOSVD. 71

6.2 Number of iterations for HOOI and trust-region algorithms for tensors $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ with elements taken from a normal distribution with zero mean and unit standard deviation. Performed are 100 runs with initial values $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$ taken from the truncated HOSVD. A run was stopped if the corresponding algorithm did not converge in 200 iterations ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$). 72

6.3 Number of inner tCG steps per trust-region step. The tensor $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ has elements taken from a normal distribution with zero mean and unit standard deviation; $R_1 = R_3 = R_3 = 2$ and $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$ are taken from the truncated HOSVD. Few tCG iterations are performed in the first trust-region steps and more in the neighborhood of the solution. . 74

6.4 Evolution of the relative norm of the gradient $\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X})$ and the cost function (2.2). The tensor $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ has elements uniformly distributed over the interval $[0,1]$; $R_1 = R_2 = R_3 = 5$. The starting point of the plots is obtained by initializing with matrices taken from the truncated HOSVD and performing 20 additional HOOI iterations. 75

6.5 Evolution of the relative norm of the gradient $\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X})$ and the cost function (2.2). The tensor $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ is as in (6.15), with $R_1 = R_2 = R_3 = 5$. The starting point of the plots is obtained by initializing with matrices taken from the truncated HOSVD and performing 5 additional HOOI iterations. 76

7.1 Illustration of the linear CG algorithm. 80

7.2 Illustration of the nonlinear CG algorithms. 82

7.3 Relative norm of the gradient $\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X})$ and cost function (2.2) for the conjugate gradient algorithm (CG) and steepest descent (SD); $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$; the multilinear rank of the approximation is $(5,5,5)$. The initial points are obtained from the truncated HOSVD. 87

7.4 A representative example and a summary of 100 runs illustrating the number of CG and HOOI iterations. The tensors $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ are with elements taken from a normal distribution with zero mean and unit standard deviation and $R_1 = R_2 = R_3 = 5$. The initial values were taken from the truncated HOSVD. 90

7.5 Summary for 100 runs of the number of iterations for CG and tCG(TR) for tensors $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ with elements taken from a normal distribution with zero mean and unit standard deviation and $R_1 = R_2 = R_3 = 5$. The initial values were taken from the truncated HOSVD. 91

7.6 Summary for 100 runs of the number of iterations for CG and tCG(TR) for tensors $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ as in (7.1) and $R_1 = R_2 = R_3 = 5$. The initial values were taken from the truncated HOSVD. The precision was equal to 10^{-6}. 92

7.7 Summary of the number of iterations for 100 runs of HOOI, tCG(TR) and CG. $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$, $R_1 = R_2 = R_3 = 5$ and the initial values were taken from the truncated HOSVD. 93

8.1 Local minima for different noise levels. The tensors $\mathscr{A} \in \mathbb{R}^{10\times 10\times 10}$ are as in (8.1)–(8.2). Performed are 100 runs starting from different arbitrary column-wise orthonormal matrices $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. A run was stopped if the corresponding algorithm did not converge ($\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations. 98

8.2 Cost function at local minima and the distance between the local minima for different σ. The tensors $\mathscr{A} \in \mathbb{R}^{10\times 10\times 10}$ are as in (8.1)–(8.2). Performed are 100 runs for each σ, starting from different arbitrary column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. Each run was stopped if the corresponding algorithm did not converge ($\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations. 99

8.3 The largest three singular values in each mode for different σ. The tensors $\mathscr{A} \in \mathbb{R}^{10\times 10\times 10}$ are as in (8.1)–(8.2). Performed are 100 runs for each σ, starting from different arbitrary column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. Each run was stopped if the corresponding algorithm did not converge ($\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations. 100

8.4 Local minima for different multilinear ranks. The entries of the tensor $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ are taken from a normal distribution with zero mean and unit standard deviation. Performed are 100 runs starting from different arbitrary column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. Each run was stopped if the corresponding algorithm did not converge ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations. 101

8.5 Local minima, corresponding to Table 8.1. The tensor $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ had elements taken from a normal distribution with zero mean and unit standard deviation. The imposed multilinear rank was $(2,2,2)$. Performed were 10 runs starting from different arbitrary column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. A run was stopped if the algorithm did not converge ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 100 iterations. The maximum number of iterations was not reached for the trust-region algorithm in any of the runs. HOOI converged for less than 100 iterations only in run 3. . . 103

8.6 Local minima for different noise levels. The tensors $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ are as in (8.1)–(8.2). Performed are 100 runs starting from different arbitrary column-wise orthonormal matrices $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. A run was stopped if the corresponding algorithm did not converge ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations. The "true" value corresponds to the subspaces spanned by the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ from the truncated HOSVD of $\mathscr{T}$. . 104

8.7 Local minima as in Figure 8.4. Additional results for HOOI and the trust-region algorithm when started from the truncated HOSVD are presented. . 105

9.1 The effect of reducing the mode-3 rank in step 3 of Algorithm 7, in the case of ill-conditioned $\mathbf{C}$, no damping. 115

9.2 The effect of reducing the mode-3 rank in step 3 of Algorithm 7, in the case of ill-conditioned $\mathbf{C}$ after introducing damping factor. 115

9.3 Condition number of the Vandermonde matrix generated by the poles in case of no damping and in case of small damping factor for one of the peaks. 116

List of Tables

8.1 Subspace angles for different matrices $\mathbf{U}$, obtained when the trust-region algorithm was initiated differently. The elements of $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ were taken from a normal distribution with zero mean and unit standard deviation. The imposed multilinear rank was $(2,2,2)$. Performed are 10 runs starting from different arbitrary column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. A run was stopped as soon as $\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$. 102

10.1 Summary of the main features of HOOI (Algorithm 1), Newton's algorithm (Algorithm 3), trust-region algorithm (Algorithm 4), conjugate gradient algorithm (Algorithm 6), Newton's algorithm [51] and quasi-Newton algorithm [118]. 121

List of Algorithms

1 Higher-order orthogonal iteration 23
2 Geometric Newton for Oja's vector field 45
3 Geometric Newton algorithm 57
4 Trust-region based algorithm 70
5 Linear conjugate gradient method 81
6 Conjugate gradient based algorithm 84
7 HO-HTLSstack 112

Chapter 1

Introduction

In this thesis, we combine two hot topics: *multilinear algebra* and *optimization on manifolds*. Multilinear algebra deals with *higher-order tensors*, which can be thought of as generalizations of vectors and matrices to three-dimensional or even higher-dimensional tables of numbers. These structures are used in many application fields, such as statistics, signal processing and scientific computing. Efficient and reliable algorithms for manipulating these multi-way arrays are thus required. The order of the tensor can be any number, for example 1859. Such tensors are difficult to visualize unlike the third-order tensors, which can be presented as cuboids as in Figure 1.1. In many applications, tensors of high order are used. However, the differences in the properties and algorithms for third-order tensors and for tensors of order higher than three are mainly technical,

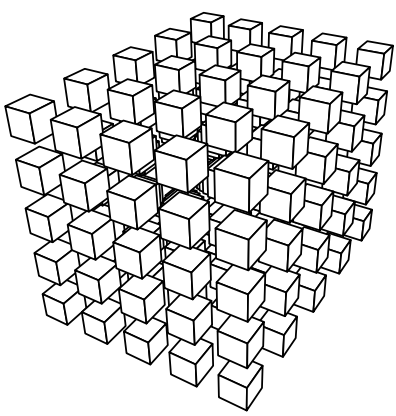


Figure 1.1: *Visualization of a third-order tensor.*

whereas the differences between the properties and algorithms for matrices and for third-order tensors are conceptual. This is why, in this thesis, we only consider third-order tensors. Generalizations of the proposed algorithms to tensors of order higher than three are feasible.

Tensor algebra is more complex, but also richer, than matrix algebra. Of main concern for this thesis is a particular generalization of the matrix rank. The rank, the row rank and the column rank of a matrix are equivalent whereas in multilinear algebra these are in general different. There are two main definitions for rank of a higher-order tensor, each one preserving some of the properties of the matrix rank. This thesis is devoted to the *multilinear rank*, as defined in [65, 66] as a generalization of column and row rank. In particular, we develop algorithms for *the best low multilinear rank approximation* of a given higher-order tensor. This means that we look for another higher-order tensor, as close as possible to the original one and such that it has multilinear rank bounded by prespecified numbers. This will be further referred to as our main tensor approximation problem. In the matrix case, the solution of such a problem is given by the truncated singular value decomposition (SVD) [57, §2.5]. In multilinear algebra, the truncated higher-order SVD (HOSVD) [36] gives a suboptimal approximation, which can be refined by iterative algorithms. The traditional algorithm for this purpose is the higher-order orthogonal iteration (HOOI) [37, 83, 84], further optimized in [12]. In this thesis, rather than trying to optimize this algorithm with linear convergence, we aim to develop conceptually faster algorithms based on the Newton method, trust-region scheme and conjugate gradients.

It appears that the function that we need to optimize in order to compute the best low multilinear rank approximation of a higher-order tensor has an undesired symmetry property by the action of the orthogonal group. Conceptually speaking, the solutions are not isolated, i.e., there are whole groups of infinitely many elements that are equivalent in some sense. This is a potential obstacle for algorithms since in practice, convergence to one particular point has to be achieved. This is where the second topic of the thesis comes into play. We will use differential geometric techniques to remove the mentioned symmetry. We choose to work on *quotient manifolds*. The elements of such spaces are sets, each set containing a point together with all other points that are equivalent to it. These sets are disjoint and their union covers the whole considered space. For our particular problem, we work with matrices that are easily represented in computer algorithms but in practice we are only interested in the subspaces that their columns span. There are infinitely many matrices with the same column space and these can be combined in one compound element of a quotient space. Another possibility is to first restrict the set of all considered matrices to the set of matrices with column-wise orthonormal columns and then combine all equivalent matrices from the selected ones in one element. This is justified by the fact that any subspace can be represented by the column space of a column-wise orthonormal matrix. In this thesis, we consider both options. This will be further explained in Chapter 3. We refer to [5] for the general theory on optimization on matrix manifolds.

1.1 Contributions of the thesis

The main contributions of this thesis are in the form of developed algorithms for computing the best low multilinear rank approximation of higher-order tensors. We proposed three algorithms and have studied their numerical properties and compared them with the existing higher-order orthogonal iteration and with each other. These new methods are

1. **Algorithm 3**: Geometric Newton algorithm,
2. **Algorithm 4**: Trust-region based algorithm,
3. **Algorithm 6**: Conjugate gradient based algorithm.

Our research work with respect to these algorithms from theory to extensive validation has been described in 5 publications [70, 72, 69, 71, 67].

The development of the Newton-type method was far from straightforward. We first had to consider an independent problem in the matrix case that had the same symmetry issues as our main problem in the tensor case. The developed machinery was then used for developing the geometric Newton algorithm for solving the main problem of the thesis. The matrix algorithm is however a meaningful algorithm on its own, summarized in

4. **Algorithm 2**: Geometric Newton for Oja's vector field.

The work has been published in [3]. The main author P.-A. Absil initiated and developed the algorithm. My main contributions are included in [3, §5].

We also mention some more general contributions. Independently of the considered algorithm, the best low multilinear rank approximation problem may have several local minima. We have analyzed this issue and discovered that it can have considerable consequences for whole classes of applications. We have also worked on one application of the best low multilinear rank approximation problem. We have investigated the issue of choosing the best values for the multilinear rank of the approximation in a parameter estimation problem. This study was done within the framework of harmonic analysis and resulted in an additional publication [73]. We summarize the last two contributions as follows:

5. **Analysis** of the issue of local minima of the best low multilinear rank approximation problem,
6. **Comparison** of the performance of matrix and tensor based multi-channel harmonic analysis.

1.2 Chapter-by-chapter overview

This thesis is divided into four main parts. In the first part, Chapters 2–3, we provide background material on multilinear algebra and optimization on manifolds, necessary for understanding the rest of the thesis.

In the second part, Chapters 4–7, we develop four algorithms. The first one is for a matrix problem with similar symmetry property as the main tensor problem. The other three algorithms are iterative algorithms for the best low multilinear rank approximation of higher-order tensors, which is the main problem in this thesis.

In the third part, Chapters 8–9, we discuss the issue of local minima and its consequences for applications. Finally, we consider one application area and compare matrix and tensor based algorithms for the same parameter estimation problem.

The last part contains our conclusions, the bibliographic references and some additional material.

Chapter 2: Multilinear algebra

This is the first of two introductory chapters. We introduce basic concepts of multilinear algebra that will be used throughout the thesis. We formulate and motivate the main problem of the thesis and summarize related methods that appeared in the literature.

The basic quantities considered here are *higher-order tensors*. We define them and provide some of their main properties. Special attention is paid to the generalization of the concept of matrix rank to higher-order tensors. We also define matrix representations of higher-order tensors and some basic operations, such as mode-n tensor-matrix product, scalar product and norm. The notation used throughout the thesis is also explained.

Next we present intuitively and formally the main problem of the thesis. This is the best low multilinear rank approximation of a higher-order tensor. Two equivalent mathematical formulations are provided, the first being more intuitive, and the second being more appropriate for further investigation. The problem has various applications. Some of them are mentioned and references are provided. One particular type of application is further discussed in Chapter 9.

We conclude the chapter with a brief presentation of a generalization of the matrix SVD to higher-order tensors, the higher-order SVD, and an overview of the higher-order orthogonal iteration, which is currently the most frequently used method for solving the approximation problem. Other related methods in the literature are also mentioned.

Chapter 3: Matrix manifolds

This chapter is the second introductory chapter. Here we introduce basic concepts of optimization on manifolds, consider two quotient manifolds in particular and motivate our choice for these complex structures.

We present the so-called *manifolds*, a concept of increasing interest in numerous application areas. The motivation for considering these non-Euclidean spaces is a symmetry property of our main cost function defined in Chapter 2. This symmetry causes problems for numerical algorithms. Working on manifolds, these problems can be effectively removed.

We give some insight in the main elements of differential geometry. In particular, we discuss *quotient manifolds*. Special attention is paid to the Grassmann manifold and to a manifold denoted by $\mathbb{R}_*^{n\times p}/O_p$. These two manifolds will be used in the algorithms developed in Chapters 4–7. Finally, we introduce the *retraction* and *vector transport*, which are improvements of the more standard *exponential map* and *parallel transport*. These improvements are crucial for developing efficient algorithms on manifolds.

Chapter 4: A geometric Newton method for Oja's vector field

In this chapter, we develop a geometric Newton algorithm for solving the matrix equation $F(\mathbf{X}) \equiv \mathbf{AX} - \mathbf{XX}^T\mathbf{AX} = \mathbf{0}$, related to Oja's vector field.

Before starting with the tensor problem, we first consider the symmetry property of the cost function in a simpler context, namely the matrix case. A simple Newton method for solving the matrix equation $F(\mathbf{X}) \equiv \mathbf{AX} - \mathbf{XX}^T\mathbf{AX} = \mathbf{0}$ faces the problem of nonisolated zeros. We exploit differential-geometric techniques to remove this symmetry and obtain a "geometric" Newton algorithm. The improved behavior of the algorithm is illustrated with numerical examples.

Chapter 5: Geometric Newton algorithm

In this chapter, we develop a geometric Newton algorithm for the best low multilinear rank approximation of tensors.

We reformulate the main problem in terms of the computation of zeros of a well-chosen cost function. The new function still has an invariance property with respect to right multiplication by orthogonal matrices. Based on the technique proposed in Chapter 4, we develop a differential-geometric Newton algorithm for our tensor optimization problem. At the end of the chapter we illustrate our numerical experience with the algorithm and compare it to the higher-order orthogonal iteration. We also discuss the computational cost and the convergence properties of the proposed algorithm.

Chapter 6: Trust-region algorithm

In this chapter, we develop a trust-region based algorithm for the main optimization problem of the thesis.

We first summarize the main idea behind the trust-region scheme and its generalization to a Riemannian manifold. Then we obtain closed-form expressions for all necessary components in order to translate our best low multilinear rank approximation problem into the framework of the trust-region algorithm on manifolds. Again, we provide numerical experiments, compare with the two algorithms that have been proposed so far and discuss the convergence speed and computational cost of the algorithm.

Chapter 7: Conjugate gradient algorithm

This chapter presents an algorithm based on conjugate gradients for solving the same tensor approximation problem as in the previous two chapters.

We start with recalling the linear and nonlinear conjugate gradient methods and their advantages and disadvantages. A generalization to an algorithm on a manifold is not straightforward and needs the notion of parallel translation, or more efficiently, a vector transport. In the conjugate gradient based tensor algorithm, many parameters need to be fixed in an efficient way. Analytical convergence properties are difficult to prove. In our numerical experiments we compare the algorithm with the previously discussed algorithms. We comment on the precision of the algorithm and possible “restarts”. We also list some advantages and disadvantages of the algorithm and suggest in which cases it performs best.

Chapter 8: Local minima

In this chapter, we study the issue of local minima of our main problem.

We illustrate the existence of local minima and look at their properties and for dependencies on the given problem. We provide some explanations and try to foresee possible consequences for real applications. Finally, we discuss the most popular initial value for the iterations, which is the truncated HOSVD.

To the best of our knowledge, the issue of local minima of the best low multilinear rank approximation has not been discussed in the literature. However, we see the existence of local minima as an important result since the corresponding problem in the matrix case has (in case of distinct singular values) only one local minimum which is the global minimum.

Chapter 9: Application: Harmonic analysis

In this chapter, we compare a matrix and a tensor based algorithm, based on the low multilinear rank approximation, for a parameter estimation problem.

We consider a decomposition of a signal in exponential components in the multi-channel case. The signal is provided as a set of simultaneous signals with identical poles but different complex amplitudes. The goal is to estimate the poles of the model, given only a limited number of samples. Matrix-based and tensor-based methods for the estimation of the unknown parameters are known, the latter using the best low multilinear rank approximation of tensors.

We compare the performance of the two types of methods for the case of closely spaced poles. We are interested in factors that improve the performance of the tensor method, in particular connected with the right choice of multilinear rank for the approximation. The aim is to give a more clear view under which conditions the tensor-based algorithm outperforms the simpler matrix-based algorithm.

Chapter 10: Conclusions and open problems

This is a concluding chapter. We summarize all obtained results and suggest directions for further research.

Part I

Background material

Chapter 2

Multilinear algebra

This chapter presents an overview of the basic concepts of multilinear algebra and introduces and motivates the main problem of this thesis.

After defining the main elements in multilinear algebra, we introduce the tools for treating them and list their main properties. Even the generalization of basic concepts from linear to multilinear algebra is not straightforward. One example is the rank of a matrix, which can be generalized in several different ways, keeping some properties of the matrix rank and leaving other properties out. We discuss two main generalizations, namely the rank and the multilinear rank. The latter is of central interest for this thesis. We formulate the main problem dealt with in the thesis and motivate our choice. Finally, we list two widely used algorithms for initialization and for treating the problem and mention related decompositions and algorithms from the literature. In this thesis, we are interested in developing alternative and possibly faster methods than the ones that are already available.

2.1 Basic definitions, operations on and properties of higher-order tensors

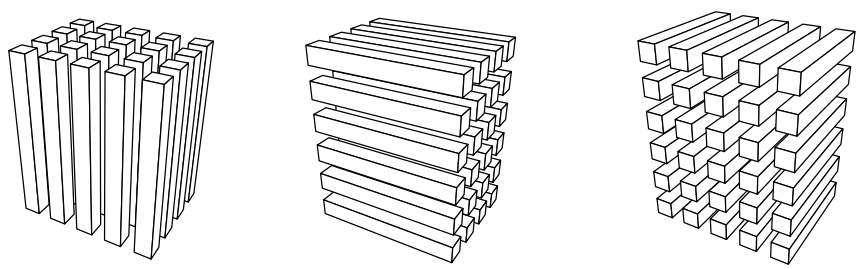
In this section, we define our main objects, present some tools for treating them and discuss some of their main properties. The objects are called higher-order tensors. These are higher-order generalizations of vectors and matrices. Two of their main characteristics are their rank and their multilinear rank, defined in Section 2.1.2. We then explain how to perform a tensor-matrix multiplication, how to represent a higher-order tensor as a matrix, and how to compute an inner product of two higher-order tensors or their norm. Finally we mention some useful properties.

2.1.1 Higher-order tensor

Consider the following sequence,

$$\underset{\text{scalar}}{a} \quad , \quad \underset{\text{vector}}{\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}} \quad , \quad \underset{\text{matrix}}{\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}} \quad , \quad \dots$$

The next element in this sequence is a *third-order tensor*, the following is a fourth-order tensor and so on. In general, the term *Nth-order tensor* is used, or *higher-order tensor* if the order is three or higher. The elements of an Nth-order tensor are referred to by N indices. Thus, higher-order tensors are multi-way arrays of numbers. In this sense, vectors are first-order and matrices are second-order tensors. Each "direction" of a higher-order tensor is called a *mode*. In the above sequence, the number of elements in each mode is 2 but in general, this number can be higher. The mode-1 vectors of a higher-order tensor are defined as its columns and the mode-2 vectors are its rows. In general, the *mode-n vectors* $(n = 1, 2, \dots, N)$ are the vectors, obtained by varying the n^{th} index, while keeping the other indices fixed. The mode-n vectors of a $(6 \times 5 \times 4)$-tensor, i.e., a third-order tensor $\mathscr{A} \in \mathbb{R}^{6\times5\times4}$, are visualized in Figure 2.1.



(a) Mode-1 vectors (b) Mode-2 vectors (c) Mode-3 vectors

Figure 2.1: *Mode-n vectors of a* $(6 \times 5 \times 4)$*-tensor.*

Higher-order tensors have various names in the literature, such as "N-way array" and "N-dimensional hypermatrix". Mode-n vectors are sometimes called fibers. A higher-order tensor is actually an element of the tensor product of N vector spaces. When the choice of basis is implicit, we think of a tensor as its representation as an N-way array [42]. However, other types of tensors are used in physics, called "tensor fields" in mathematics, and these should not be confused with the higher-order tensors discussed in this thesis. An overview paper on higher-order tensors, tensor decompositions and some applications has recently been published as [82]. In this thesis, we stick to the definitions presented in the current chapter.

Throughout the thesis, we use the following notation. Scalars are denoted as lower-case letters ($a, b, \ldots, \alpha, \beta, \ldots$), vectors are written as bold-face letters ($\mathbf{a}, \mathbf{b}, \ldots$), bold-face capitals ($\mathbf{A}, \mathbf{B}, \ldots$) correspond to matrices and tensors are denoted by calligraphic letters ($\mathscr{A}, \mathscr{B}, \ldots$). Thus, the elements of a third-order tensor $\mathscr{A}$ are $a_{ijk} = (\mathscr{A})_{ijk}$. Some special scalars and upper bounds are denoted by capital letters ($I, I_1, I_2, I_3, N, R, \ldots$). We use "$\times$" for the Cartesian product of sets and "$\otimes$" for the Kronecker product. The identity matrix is $\mathbf{I}$, $O_p = \{\mathbf{X} \in \mathbb{R}^{p \times p} : \mathbf{X}^T\mathbf{X} = \mathbf{X}\mathbf{X}^T = \mathbf{I}\}$ denotes the orthogonal group (the set of all orthogonal $(p \times p)$-matrices), and $St(p,n) = \{\mathbf{X} \in \mathbb{R}^{n \times p} : \mathbf{X}^T\mathbf{X} = \mathbf{I}, n \geq p\}$ denotes the Stiefel manifold (the set of all column-wise orthonormal $(n \times p)$-matrices).

2.1.2 Rank and multilinear rank

Generalization of matrices to higher-order tensors is straightforward. However, matrix properties, even basic ones, are not uniquely generalizable to higher-order tensors. Here, we discuss the rank of a matrix. Different matrix-rank properties lead to different tensor-rank definitions. We first consider a definition of tensor-rank introduced in [65, 66] as a direct generalization of row and column rank of a matrix. The *mode-n rank* of a tensor $\mathscr{A}$, denoted by $\text{rank}_n(\mathscr{A})$, is the number of linearly independent mode-n vectors of the tensor. Contrary to the matrix case, different mode-n ranks are in general different from each other. The *multilinear rank* of a tensor is then the n-tuple of the mode-n ranks, i.e, $(R_1, R_2, \ldots, R_N)$, where $R_n, n = 1, 2, \ldots, N$ are the mode-n ranks.

In the second-order case, the rank of a matrix also corresponds to the minimal number of rank-1 matrices that sum up to the original matrix. It is also possible to generalize this idea to tensors. For this purpose, we first define a rank-1 tensor. A *rank-1 tensor* is an outer product of N nonzero vectors. For example, for a third-order rank-1 tensor, the element at position (i, j, k) is the product $a_i b_j c_k$, where $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are the three vectors in question. This concept is visualized in Figure 2.2. The *rank* of a higher-order tensor is

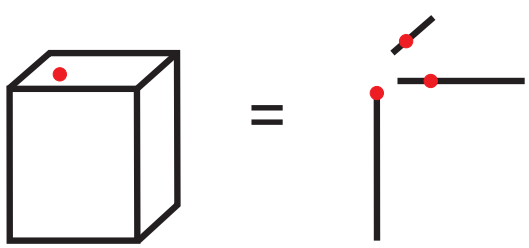


Figure 2.2: *Rank-1 tensor as an outer product of three vectors.*

the minimal number of rank-1 tensors that sum up to the original tensor [65, 66, 85]. In general, the multilinear rank is different from the rank of the higher-order tensor.

In this thesis, we concentrate on the multilinear rank of the tensors and from now on, we will mainly consider third-order tensors. Generalizations to higher-order tensors of the theory that is proposed in this thesis is technical and the principles stay the same. We will often only write "tensor" when we mean a third-order or a higher-order tensor.

2.1.3 Basic operations

We start this section by defining the product of a tensor and a matrix.

Definition The *mode-n products* $\mathscr{A} \bullet_n \mathbf{M}^{(n)}, n = 1,2,3$ of a tensor $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ with matrices $\mathbf{M}^{(n)} \in \mathbb{R}^{J_n \times I_n}$ are defined by

$$
\begin{aligned}
(\mathscr{A} \bullet_1 \mathbf{M}^{(1)})_{j_1 i_2 i_3} &= \textstyle\sum_{i_1} a_{i_1 i_2 i_3} m^{(1)}_{j_1 i_1}, \\
(\mathscr{A} \bullet_2 \mathbf{M}^{(2)})_{i_1 j_2 i_3} &= \textstyle\sum_{i_2} a_{i_1 i_2 i_3} m^{(2)}_{j_2 i_2}, \\
(\mathscr{A} \bullet_3 \mathbf{M}^{(3)})_{i_1 i_2 j_3} &= \textstyle\sum_{i_3} a_{i_1 i_2 i_3} m^{(3)}_{j_3 i_3},
\end{aligned}
$$

where $1 \leq i_n \leq I_n$, $1 \leq j_n \leq J_n$.

In this notation, the product of a matrix $\mathbf{M}$ with a matrix $\mathbf{U}$ from the left and a matrix $\mathbf{V}^T$ from the right, i.e., $\mathbf{A} = \mathbf{U}\mathbf{M}\mathbf{V}^T$ is presented as $\mathbf{A} = \mathbf{M} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V}$. This presentation is reasonable since the columns of $\mathbf{U}$ correspond to the column space of $\mathbf{A}$ in the same way as the columns of $\mathbf{V}$ correspond to the row space of $\mathbf{A}$. Thus, the relationship between $\mathbf{M}$ and $\mathbf{U}$ is similar to the one between $\mathbf{M}$ and $\mathbf{V}$. Some properties of the mode-*n* product are given in the next section.

Often, it is useful to reorganize the elements of a higher-order tensor in a matrix form. To do so, we first choose one of the modes of the higher-order tensor, say *n*. Then, all the mode-*n* vectors are put one after the other in a specific order. We use the following definition.

Definition The *matrix representations* $\mathbf{A}_{(n)}, n = 1,2,3$ of a tensor $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ are defined as follows

$$
(\mathbf{A}_{(1)})_{i_1,(i_2-1)I_3+i_3} = (\mathbf{A}_{(2)})_{i_2,(i_3-1)I_1+i_1} = (\mathbf{A}_{(3)})_{i_3,(i_1-1)I_2+i_2} = a_{i_1 i_2 i_3},
$$

where $1 \leq i_n \leq I_n$.

Sometimes they are also called unfoldings or flattenings. The definition is illustrated in Figure 2.3 for the case $I_1 > I_2 > I_3$. Different definitions of matrix representations are

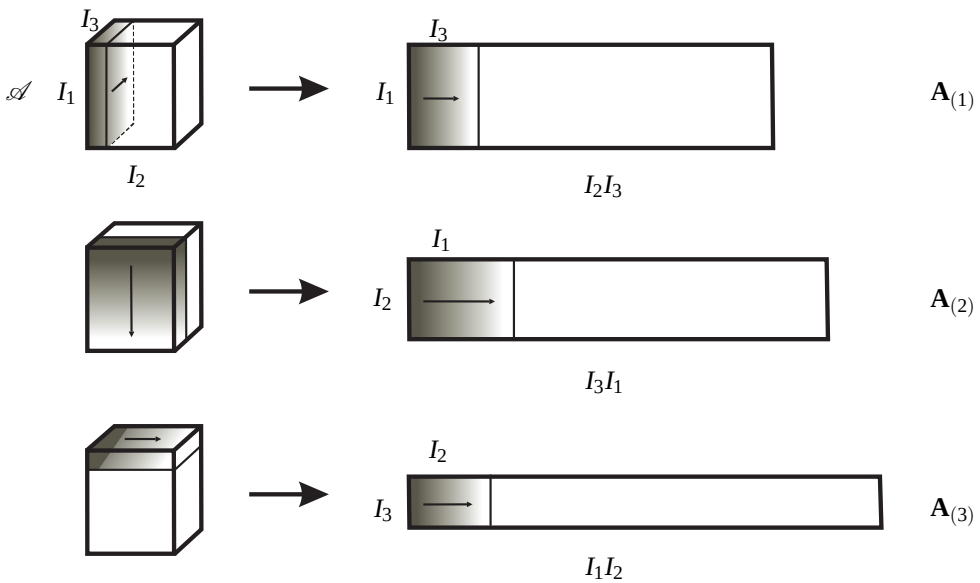


Figure 2.3: *Matrix representations of a tensor.*

used in the literature. However, the properties of the tensors are independent of these representations, although the particular formulas may differ.

Finally, we give definitions for the scalar product of two tensors and the Frobenius norm of a tensor.

Definition The *scalar product* of two tensors $\mathscr{A}, \mathscr{B} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is defined as

$$\langle \mathscr{A}, \mathscr{B} \rangle = \sum_{i_1} \sum_{i_2} \sum_{i_3} a_{i_1 i_2 i_3} \, b_{i_1 i_2 i_3}.$$

Definition The *Frobenius norm* of a tensor $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is defined as

$$\|\mathscr{A}\| = \sqrt{\langle \mathscr{A}, \mathscr{A} \rangle}.$$

2.1.4 Main properties

We first mention some properties of the rank and the multilinear rank of a tensor $\mathscr{A}$. The mode-n rank $\text{rank}_n(\mathscr{A})$ can be obtained from the mode-n matrix representation of $\mathscr{A}$ in the following way

$$\text{rank}_n(\mathscr{A}) = \text{rank}(\mathbf{A}_{(n)}).$$

Further, given a tensor with multilinear rank equal to (R_1, R_2, R_3), the following inequalities hold for the mode-n ranks.

$$R_n \leq \prod_{i \neq n} R_i, \; n = 1,2,3.$$

There is no closed-form formula for the rank of a tensor $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$; there are only some partial results for special types of tensors (see [82, 42] and the references therein). The rank is not even bounded by $\min(I_1, I_2, I_3)$. Moreover, if $\mathscr{A}$ is considered as a tensor in the complex field, its rank could be different from the rank of the same tensor considered in $\mathbb{R}^{I_1 \times I_2 \times I_3}$ [86]. In this thesis, we consider the multilinear rank of real-valued tensors. Generalizations to complex tensors are possible but mainly technical. The multilinear rank concept is less sensitive to differences in the considered field, as it is closer to matrix algebra.

The following inequality gives a relation between the rank and the multilinear rank

$$R_n \leq R, \quad n = 1,2,3.$$

Concerning the mode-n product, we will often use the following associativity properties

$$(\mathscr{A} \bullet_n \mathbf{U}) \bullet_m \mathbf{V} = (\mathscr{A} \bullet_m \mathbf{V}) \bullet_n \mathbf{U} = \mathscr{A} \bullet_n \mathbf{U} \bullet_m \mathbf{V}, \quad m \neq n,$$

$$(\mathscr{A} \bullet_n \mathbf{U}) \bullet_n \mathbf{V} = \mathscr{A} \bullet_n (\mathbf{V}\mathbf{U}).$$

Finally, we show how tensor-matrix products can be reformulated using matrix representations, namely,

$$\begin{aligned} \|\mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T\|^2 &= \|\mathbf{U}^T \mathbf{A}_{(1)} (\mathbf{V} \otimes \mathbf{W})\|^2 \\ &= \|\mathbf{V}^T \mathbf{A}_{(2)} (\mathbf{W} \otimes \mathbf{U})\|^2 \\ &= \|\mathbf{W}^T \mathbf{A}_{(3)} (\mathbf{U} \otimes \mathbf{V})\|^2. \end{aligned} \tag{2.1}$$

These reformulations will prove to be useful when developing algorithms and discussing some particular properties of the problem presented in the next section. For more detailed information on the material presented in Section 2.1.3 and Section 2.1.4, we refer the reader to [30].

2.2 Best low multilinear rank approximation

The main concern of this thesis is the best multilinear rank approximation of a given tensor. The solution is another tensor, as close as possible to the original one and having mode-n ranks bounded by prespecified R_1, R_2, R_3, respectively. It is a possible generalization of the best rank-R approximation problem of a matrix.

In mathematical terms, given a tensor $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, its best rank-$(R_1, R_2, R_3)$ approximation is a tensor $\hat{\mathscr{A}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, such that it *minimizes* the least-squares cost function $f : \mathbb{R}^{I_1 \times I_2 \times I_3} \to \mathbb{R}$,

$$f : \hat{\mathscr{A}} \mapsto \|\mathscr{A} - \hat{\mathscr{A}}\|^2 \tag{2.2}$$

under the constraints $\text{rank}_1(\hat{\mathscr{A}}) \leq R_1$, $\text{rank}_2(\hat{\mathscr{A}}) \leq R_2$, $\text{rank}_3(\hat{\mathscr{A}}) \leq R_3$.

It has been shown in [37, 83, 84] that problem (2.2) is equivalent to the problem of *maximizing* the function

$$\begin{aligned} &\overline{g} : St(R_1, I_1) \times St(R_2, I_2) \times St(R_3, I_3) \to \mathbb{R}, \\ &\quad (\mathbf{U}, \mathbf{V}, \mathbf{W}) \mapsto \|\mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T\|^2 = \|\mathbf{U}^T \mathbf{A}_{(1)} (\mathbf{V} \otimes \mathbf{W})\|^2 \end{aligned} \tag{2.3}$$

over the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ (recall that $St(p,n)$ stands for the set of column-wise orthonormal $(n \times p)$-matrices). This fact is not surprising since it is a direct generalization [41] of the matrix case. There, given a matrix $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2}$, the problem of finding the best rank-R approximation $\hat{\mathbf{A}} = \mathbf{U}\,\mathbf{B}\,\mathbf{V}^T$ with $\mathbf{B} \in \mathbb{R}^{R \times R}$ and column-wise orthonormal $\mathbf{U} \in \mathbb{R}^{I_1 \times R}$ and $\mathbf{V} \in \mathbb{R}^{I_2 \times R}$ is equivalent to the maximization of $\|\mathbf{U}^T\,\mathbf{A}\,\mathbf{V}\| = \|\mathbf{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T\|$.

In this thesis, our goal is to solve the maximization problem (2.3). In order to optimize (2.2), it is sufficient to determine $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ in (2.3). Having estimated these matrices, the optimal tensor $\hat{\mathscr{A}}$ is computed by

$$\hat{\mathscr{A}} = \mathscr{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W},$$

where $\mathscr{B} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is given by

$$\mathscr{B} = \mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T.$$

The best multilinear rank approximation problem is more complicated than the best rank-R approximation of a matrix and cannot be computed in a straightforward way. A good starting point is given by the truncated higher-order singular value decomposition, discussed in Section 2.4.1. This initial approximation can then be further improved by iterative algorithms.

2.3 Motivation

Higher-order tensors are used as a tool in higher-order statistics [101, 95, 130, 100], independent component analysis [34, 35, 41, 24] and harmonic analysis [111, 112]. Fields of application of higher-order tensors also include chemometrics [123], biomedical signal processing [99, 43, 44, 8], telecommunications [120, 121, 34], scientific computing [19, 18, 59, 109] and image processing [140]. For more references and application areas, we refer to the books [123, 83], to the overview paper on tensor decompositions and their applications [82] and to the references therein.

The main applications of the best low multilinear rank approximation of tensors are *dimensionality reduction* of tensors with high dimensions [41, 8, 43, 44, 83, 123, 82] and *signal subspace estimation* [111, 112, 83, 123, 82, 58], discussed next in some more detail.

Independent component analysis (ICA) [41] in fields like electro-encephalography (EEG), magneto-encephalography (MEG), nuclear magnetic resonance (NMR), extracts statistically independent sources from a linear mixture. Sometimes only a few sources have significant contributions. A principal component analysis (PCA)-based prewhitening step for reducing the dimensionality is often used. This approach is beneficial if the data are affected by white Gaussian noise but it is not applicable anymore if colored Gaussian noise is present. In such cases, low multilinear rank approximation of a higher-order cumulant tensor of the observation vector can be performed instead. The dimensionality of the problem is then reduced from the number of observation channels to the number of sources.

Another application of the best low multilinear rank approximation problems is simultaneous dimensionality reduction of a matrix and a tensor. This application was also discussed in [41].

Parallel factor decomposition (PARAFAC) [61], also called canonical decomposition (CANDECOMP) [21], is a decomposition of higher-order tensors in rank-1 terms. It is widely used in chemometrics [123] and has several applications in wireless communication [120, 34]. PARAFAC can also be used for epileptic seizure onset localization [44, 43, 8], since only one of its components is related to the seizure activity. The best low multilinear rank approximation of higher-order tensors is often used as a preprocessing dimensionality reduction step preceding the actual computation of PARAFAC. Such a preprocessing step is implemented for example in the *N*-way toolbox for MATLAB [13].

Dimensionality reduction works as illustrated in Figure 2.4. See also [30, Remark 6.2.2]. Let the rank-R decomposition of $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ be required. If $R < \max(I_1, I_2, I_3)$, then a reduction of $\mathscr{A}$ to a tensor $\mathscr{B} \in \mathbb{R}^{I_1' \times I_2' \times I_3'}$, $I_n' = \min(I_n, R), n = 1,2,3$ can be used for the actual computation of PARAFAC. This can be done as follows. Let $\hat{\mathscr{A}}$ be the best

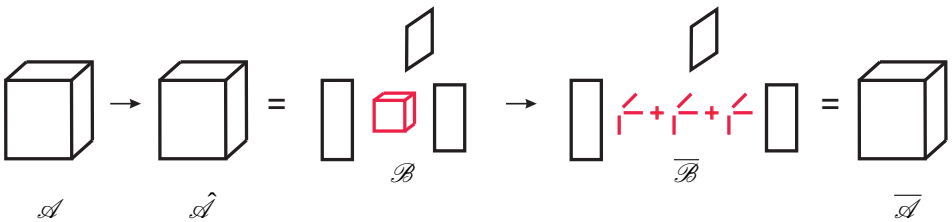


Figure 2.4: *Dimensionality reduction.*

rank-(I_1', I_2', I_3') approximation of $\mathscr{A}$. If $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are the matrices as in (2.3), i.e., if

$$\hat{\mathscr{A}} = \mathscr{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}$$

then a rank-R approximation $\overline{\mathscr{A}}$ of $\mathscr{A}$ is computed from the best rank-R approximation $\overline{\mathscr{B}}$ of $\mathscr{B}$ in the following way

$$\overline{\mathscr{A}} = \overline{\mathscr{B}} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}.$$

$\mathscr{B}$ has smaller dimensions than $\mathscr{A}$ so that the computation of $\overline{\mathscr{B}}$ is much less expensive than the direct computation of $\overline{\mathscr{A}}$. In practice, due to numerical problems, in some applications $I_n' = \min(I_n, R+2), n = 1,2,3$ are used instead of the dimensions $I_n' = \min(I_n, R)$. In general, it is advisable to examine the mode-n singular values for gaps between them and use a corresponding low multilinear rank approximation. It might also be useful to perform a few additional PARAFAC steps on $\overline{\mathscr{A}}$ in order to find an even better approximation of $\mathscr{A}$.

In many signal processing applications, a signal is decomposed as a sum of exponentially damped sinusoids. Given only samples of the signal, the poles and the complex amplitudes may be estimated by means of well-known matrix methods, see [50]. Tensor-based algorithms are developed in [111, 112]. These are based on the best rank-(R_1, R_2, R_3) approximation of a tensor. We discuss this topic further in Chapter 9.

Furthermore, the best low multilinear rank approximation of tensors can be applied in image synthesis, analysis and recognition. For example, a set of facial images can be represented as a higher-order tensor, where different modes of the tensor correspond to different factors, such as face expression, position of the head relative to the camera, and illumination [140].

The best low multilinear rank approximation of tensors has also applications in scientific computing. It can be used for approximating higher-order tensors, obtained by discretizing Newton potential, Slater-type functions, Yukawa and Helmholtz potentials [78]. Finally, we mention applications in fuzzy control [16].

2.4 Higher-order singular value decomposition, higher-order orthogonal iteration and related methods

In this section we briefly present a higher-order generalization of the matrix singular value decomposition. We then present a widely used method for solving the problem defined in Section 2.2 and mention some related methods in the literature.

2.4.1 Higher-order singular value decomposition

In the sense of multilinear rank, a generalization of the singular value decomposition (SVD) [57, §2.5], giving the best low rank approximation of a matrix, is the higher-order singular value decomposition (HOSVD) [36]. The same type of decomposition but with possible variations is also known as Tucker decomposition [133, 134]. For third-order tensors, the computation of the HOSVD involves only three SVDs. Truncation of HOSVD results in a suboptimal solution of the best low multilinear rank approximation problem, which can serve as a good starting point for iterative algorithms.

We first summarize the singular value decomposition of a matrix.

Theorem 2.4.1 (Singular value decomposition)
Every matrix $\mathbf{A} \in \mathbb{R}^{I \times J}$ *can be decomposed as a product of a pseudo-diagonal matrix* $\mathbf{\Sigma} \in \mathbb{R}^{I \times J}$ *and two orthogonal matrices* $\mathbf{U} \in \mathbb{R}^{I \times I}$ *and* $\mathbf{V} \in \mathbb{R}^{J \times J}$,

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

where the diagonal elements $\sigma_1, \sigma_2, \ldots, \sigma_{min(I,J)}$ *of* $\mathbf{\Sigma}$ *are positive, real and in non-increasing order*

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{min(I,J)} \geq 0.$$

These elements are called singular values of $\mathbf{A}$*. The column vectors of* $\mathbf{U}$ *and* $\mathbf{V}$ *are the left and right singular vectors respectively.*

The rank of $\mathbf{A}$ equals the number of non-zero singular values. The best rank-R approximation of $\mathbf{A}$ is obtained by keeping the first R singular values and setting the rest to zero, also called truncation of $\mathbf{A}$.

For third-order tensors, the HOSVD is defined as follows.

Theorem 2.4.2 (Third-order singular value decomposition)
Every tensor $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ *can be decomposed as a product of a tensor* $\mathscr{S} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ *and three orthogonal matrices* $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$, $n = 1,2,3$,

$$\mathscr{A} = \mathscr{S} \bullet_1 \mathbf{U}^{(1)} \bullet_2 \mathbf{U}^{(2)} \bullet_3 \mathbf{U}^{(3)},$$

where the matrices (subtensors) $\mathbf{S}_{i_n=\alpha}$, *obtained by fixing the* n^{th} *index of* $\mathscr{S}$ *to* α *have the following two properties:*

- *all-orthogonality: the matrices* $\mathbf{S}_{i_n=\alpha}$ *and* $\mathbf{S}_{i_n=\beta}$ *are orthogonal for any* n, α *and* β, *such that* $\alpha \neq \beta$, *i.e.,*

$$\langle \mathbf{S}_{i_n=\alpha}, \mathbf{S}_{i_n=\beta} \rangle = 0, \quad \alpha \neq \beta,$$

- *ordering:*

$$\|\mathbf{S}_{i_n=1}\| \geq \|\mathbf{S}_{i_n=2}\| \geq \cdots \geq \|\mathbf{S}_{i_n=I_n}\| \geq 0, \quad \textit{for every } n.$$

The Frobenius norms $\|\mathbf{S}_{i_n=i}\|$ *are the mode-n singular values of* $\mathscr{A}$, *denoted by* $\sigma_i^{(n)}$ *and the columns of* $\mathbf{U}^{(n)}$ *are the mode-n singular vectors.* This HOSVD is a normalized representation of the Tucker decomposition, introduced in [133, 134]. For a visualization of the decomposition see Figure 2.5.

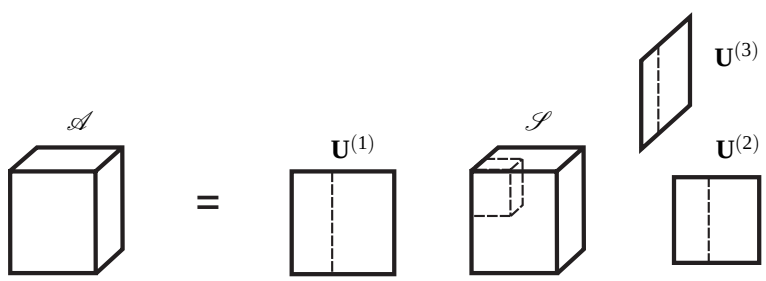


Figure 2.5: *Higher-order singular value decomposition.*

The matrix of the mode-*n* singular vectors $\mathbf{U}^{(n)}$ (and the mode-*n* singular values) of a tensor $\mathscr{A}$ can be computed as the matrix of the left singular vectors (and the singular values) of the mode-*n* matrix representation $\mathbf{A}_{(n)}$. Hence, the computation of the HOSVD of a third-order tensor $\mathscr{A}$ consists of the computation of three matrix SVDs of the matrices $\mathbf{A}_{(1)} \in \mathbb{R}^{I_1 \times I_2 I_3}$, $\mathbf{A}_{(2)} \in \mathbb{R}^{I_2 \times I_3 I_1}$, and $\mathbf{A}_{(3)} \in \mathbb{R}^{I_3 \times I_1 I_2}$. The tensor $\mathscr{S}$ can be computed through the following equality

$$\mathscr{S} = \mathscr{A} \bullet_1 \mathbf{U}^{(1)^T} \bullet_2 \mathbf{U}^{(2)^T} \bullet_3 \mathbf{U}^{(3)^T}.$$

The latter is usually a full tensor. In general, it is not possible to reduce a higher-order tensor to a pseudo-diagonal form by means of orthogonal transformations. This is due to the fact that the number of degrees of freedom in such a decomposition would be smaller than the number of the elements of the tensor that need to be decomposed. Note however that for second-order tensors, i.e., matrices, HOSVD reduces to the well-known SVD. Taking all properties of HOSVD into account, it is clear that it is a higher-order generalization [36] of SVD.

Other generalizations of the matrix SVD are also possible, focusing on different properties of the SVD. In the literature, decompositions are considered where the tensor corresponding to Σ is as diagonal as possible (in a least squares sense) [24, 38, 96, 22], or where the original tensor is decomposed in a minimal number of rank-1 terms (CANDECOMP/PARAFAC) [25, 61, 21, 39, 31], on which orthogonal [81] or symmetry [26] constraints could be imposed. Recently, block term decompositions have been proposed as the unifying framework for Tucker/HOSVD and CANDECOMP/PARAFAC [32, 33, 40].

The truncated SVD gives the best low-rank approximation of a matrix. However, the truncated HOSVD usually gives a good but not the best possible low multilinear rank approximation of a higher-order tensor. Truncation means to discard the smallest mode-n singular values for all n. The truncated HOSVD is a reasonable approximation, since the ordering property in Theorem 2.4.2 implies that "more energy" is concentrated in the part corresponding to the first singular values, i.e., the part that is left after truncation. Moreover, if there is a considerable gap between the $\sigma_{R_n}^{(n)}$ and $\sigma_{R_n+1}^{(n)}$ singular values for all n, the truncated HOSVD yields a good approximation. In mathematical terms, we have the following error bound [36].

Proposition 2.4.3 *Let $\mathscr{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ be decomposed as in Theorem 2.4.2 and let the multilinear rank of $\mathscr{A}$ be (R'_1, R'_2, R'_3). Let $\hat{\mathscr{A}}$ be obtained by replacing the smallest mode-n singular values $\sigma_{R_n+1}^{(n)}, \sigma_{R_n+2}^{(n)}, \ldots, \sigma_{R'_n}^{(n)}$ by zero for all n and for specified values R_n, $n = 1,2,3$. Then,*

$$\|\mathscr{A} - \hat{\mathscr{A}}\|^2 \leq \sum_{i_1=R_1+1}^{R'_1} \sigma_{i_1}^{(1)^2} + \sum_{i_2=R_2+1}^{R'_2} \sigma_{i_2}^{(2)^2} + \sum_{i_3=R_3+1}^{R'_3} \sigma_{i_3}^{(3)^2}.$$

Truncation may destroy the all-orthogonality but the HOSVD of the tensor $\hat{\mathscr{A}}$ can be easily computed, if necessary.

The approximation obtained by truncating the HOSVD for solving (2.3) and (2.2) respectively is often refined by iterative algorithms.

2.4.2 Higher-order orthogonal iteration

The most widely used iterative algorithm for maximizing (2.3) and thus minimizing (2.2) is the higher-order orthogonal iteration (HOOI) [37, 83, 84].

HOOI is an alternating least-squares (ALS) algorithm for the (local) minimization of $f(\hat{\mathscr{A}})$, where at each step the estimate of one of the matrices $\mathbf{U},\mathbf{V},\mathbf{W}$ is optimized, while the other two are kept constant. While optimizing with respect to the unknown orthonormal matrix $\mathbf{U}$, the function g from (2.3) is thought of as a quadratic expression in the components of $\mathbf{U}$ (the matrices $\mathbf{V},\mathbf{W}$ stay fixed). Since

$$\overline{g}(\mathbf{U},\mathbf{V},\mathbf{W}) = \|\mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T\|^2 = \|\mathbf{U}^T (\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W}))\|^2,$$

the columns of $\mathbf{U} \in \mathbb{R}^{I_1\times R_1}$ should build an orthonormal basis for the left R_1-dimensional dominant subspace of $\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})$. The solution can be obtained from the SVD of $\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})$. The optimization with respect to the other two unknown matrices $\mathbf{V}$ and $\mathbf{W}$ is performed by analogy. A summary of the algorithm is given in Algorithm 1.

Algorithm 1 Higher-order orthogonal iteration

1: Obtain initial estimates $\mathbf{U}_0 \in St(R_1,I_1)$, $\mathbf{V}_0 \in St(R_2,I_2)$, $\mathbf{W}_0 \in St(R_3,I_3)$, e.g., from the truncated HOSVD [36].
2: **for** $k = 0,1,2,\ldots$ **do**
3: Compute the R_1-dimensional left dominant subspace of $\mathbf{A}_{(1)}(\mathbf{V}_k\otimes\mathbf{W}_k)$.
As columns of $\mathbf{U}_{k+1}$, take orthonormal basis vectors of this subspace.
4: Compute the R_2-dimensional left dominant subspace of $\mathbf{A}_{(2)}(\mathbf{W}_k\otimes\mathbf{U}_{k+1})$.
As columns of $\mathbf{V}_{k+1}$, take orthonormal basis vectors of this subspace.
5: Compute the R_3-dimensional left dominant subspace of $\mathbf{A}_{(3)}(\mathbf{U}_{k+1}\otimes\mathbf{V}_{k+1})$.
As columns of $\mathbf{W}_{k+1}$, take orthonormal basis vectors of this subspace.
6: **end for**
7: $\mathbf{U} = \mathbf{U}_{k+1}, \mathbf{V} = \mathbf{V}_{k+1}, \mathbf{W} = \mathbf{W}_{k+1}$.
8: Compute

$$\mathscr{B} = \mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T,$$
$$\hat{\mathscr{A}} = \mathscr{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}.$$

The main computational cost of one iteration of HOOI comes from computing the three SVDs and forming the matrices of the form $\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})$. It is not even necessary to compute the whole SVDs but only the first left singular vectors. If we assume for simplicity that $R_1 = R_2 = R_3 = R$ and $I_1 = I_2 = I_3 = I$, then the number of flops for computing the singular vectors is approximately $3(6IR^4 + 11R^6)$ [57, §5.4.5], i.e., $O(IR^4 + R^6)$. The expression $\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})$ is a matrix representation of $\mathscr{A} \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T$. The cost for computing $\mathscr{A} \bullet_2 \mathbf{V}^T$ is of order $O(I^3R)$ (see also the definition of the mode-n

product, Section 2.1.3) and the number of flops necessary for computing $\mathscr{A} \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T$ given $\mathscr{A} \bullet_2 \mathbf{V}^T$ is $O(I^2R^2)$. Thus, the computational cost for $\mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W})$ is of order $O(I^3R)$. The total cost for one iteration of HOOI is then $O(I^3R + IR^4 + R^6)$ [51, 72].

Initial values for HOOI are often taken from the truncated HOSVD. The matrices obtained from the truncated HOSVD usually belong to the attraction region of (2.3). However, there are exceptions. Moreover, convergence to the global maximum is not guaranteed.

HOOI is a simple concept and easy to implement. Therefore it comes as no surprise that it is the most widely used algorithm at the moment [82]. However, its convergence speed is at most linear. In this thesis, we develop several algorithms for solving the same problem, aiming at a higher convergence speed.

2.4.3 Other methods in the literature

Recently, a Newton-type algorithm for the best low multilinear rank approximation of tensors has been proposed in [51]. It works on the so-called Grassmann manifold whereas the Newton-type algorithm developed in this thesis is a generalization of the ideas behind the geometric Newton method for Oja's vector field [3]. Quasi-Newton methods on Grassmannians have been suggested in [118].

We also mention other related methods that have appeared in the literature. A Krylov method has been proposed in [117] for large sparse tensors. In [37, 146, 80], specific algorithms for the best rank-1 approximation have been discussed. Fast HOSVD algorithms for symmetric, Toeplitz and Hankel tensors have been proposed in [14]. For tensors with large dimensions, Tucker-type decomposition are developed in [109, 20, 91]. In [109], it is required that a good Tucker approximation exists with low multilinear rank. The computation then involves only few entries of the given tensor and takes only linear time. In [91], a mode that is qualitatively different from all others is expected.

2.5 Conclusions

Higher-order tensors are the building blocks of multilinear algebra. In this chapter, we defined these objects and discussed their main properties. We formulated the main problem discussed in this thesis, the best low multilinear rank approximation of a tensor, and motivated our choice.

A higher-order generalization of the matrix SVD was presented that gives a good approximation of the required solution. This approximation is a suggested way of initializing iterative algorithms. One such algorithm is HOOI — a widely used method for finding a local minimum of our problem (2.3). This algorithm was summarized and some other related algorithms from the literature were mentioned.

Chapter 3

Matrix manifolds

As explained in Section 2.2, the main concern of this thesis is solving (2.3) in an optimal way. The solutions $(\mathbf{U}^*, \mathbf{V}^*, \mathbf{W}^*)$ lie on a manifold that is a product of three Stiefel manifolds. Recall that the Stiefel manifold $St(p,n)$ stands for the set of column-wise orthonormal $(n \times p)$-matrices. Due to an invariance property of the cost function (2.3), the solutions are not isolated which might cause difficulties for numerical methods (see, for example, [5, Prop. 2.1.2] or [3]). A solution to this problem is to combine equivalent solutions in one element in a certain way and work on the obtained quotient manifold.

In this chapter, we first explain the invariance property in question. We then give some basic definitions of manifolds and their main elements. We discuss quotient manifolds in particular. Special attention is paid to the Grassmann manifold and the manifold $\mathbb{R}_*^{n\times p}/O_p$ since they will be used further in the thesis. Finally, we illustrate the two notions of retraction and vector transport that make it possible to represent behavior on a matrix manifold using standard matrix algebra computations. This chapter is to a large extent based on [5, 71, 3, 72]. General references on optimization on manifolds include [54, 124, 125, 135, 93, 92, 48, 9, 1].

In this thesis, we use manifolds as a tool rather than that we work on their theoretical properties. We do not present the exhaustive theory but we only present definitions, notions and properties that are necessary for the developments of our algorithms in the following chapters. In the usual trade-off between being precise and giving the intuition, we lean towards the latter and refer the reader to the literature for more details. For the general theory on optimization on matrix manifolds we refer to [5].

3.1 Motivation: Invariance property

For third-order tensors, the cost function $\bar{g}$ from (2.3) can be computed as

$$\begin{aligned}\bar{g}(\mathbf{U},\mathbf{V},\mathbf{W}) = \|\mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T\|^2 &= \|\mathbf{U}^T\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})\|^2 \\ &= \|\mathbf{V}^T\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})\|^2 \\ &= \|\mathbf{W}^T\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})\|^2.\end{aligned}$$

Notice that $\bar{g}$ has the following invariance property

$$\bar{g}(\mathbf{U},\mathbf{V},\mathbf{W}) = \bar{g}(\mathbf{U}\mathbf{Q}^{(1)},\mathbf{V}\mathbf{Q}^{(2)},\mathbf{W}\mathbf{Q}^{(3)}), \tag{3.1}$$

where $\mathbf{Q}^{(i)} \in O_{R_i}, i = 1,2,3$ are orthogonal matrices. This means that we are not interested in the exact elements of the matrices $\mathbf{U},\mathbf{V},\mathbf{W}$ but in the subspaces that their columns span.

A consequence of (3.1) is that the solutions of $\bar{g}$ are not isolated. Thus, numerical methods might have difficulties converging to a solution (see, for example, [5, Prop. 2.1.2] or [3]). We remedy this problem using a differential-geometric approach. The necessary basics for this strategy are summarized in the current chapter.

3.2 General concepts

Constrained optimization and optimization on manifolds are related in the sense that in optimization on manifolds the constraints are moved from the problem to the search space, i.e., the manifold.

In general, a *manifold* can be thought of as a smooth surface. It is a space that can locally be represented by a Euclidean space (vector space with an assigned metric). For example, a plane and a sphere are two simple manifolds. Globally, a manifold can be a very complicated structure but it can always be divided in smaller regions that cover the whole manifold with possible overlap. There is a one-to-one correspondence between each such region and an open Euclidean set. The dimension of the open sets is also the dimension of the manifold. It is important that if a point is covered by more than one region, the corresponding maps are compatible.

We will work on *differentiable manifolds*. These manifolds allow defining functions that are differentiable in some sense, as well as defining tangent vectors. The set of all tangent vectors at a given point on the manifold is its *tangent plane*, also called *tangent space*. A visualization of the tangent plane $T_{\mathbf{X}}M$ at a point $\mathbf{X}$ of the manifold M is provided in Figure 3.1. The vector $\xi \in T_{\mathbf{X}}M$ is a tangent vector at $\mathbf{X} \in M$. In particular, we will work on *Riemannian manifolds*. These are differentiable manifolds with an associated inner product in each tangent space. The inner product allows us to consider length, distances, angles and gradients of functions.

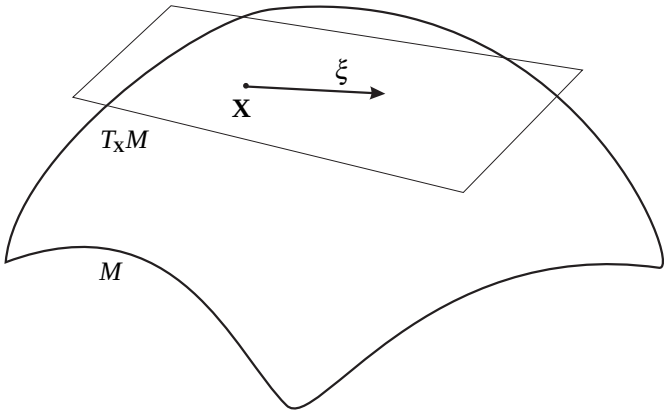


Figure 3.1: *Manifold M, tangent plane $T_X M$ and a tangent vector ξ at point $\mathbf{X} \in M$.*

There are two main approaches to reduce the set of solutions to isolated solutions only. It is possible to impose an additional constraint on the iterates produced by the numerical algorithm, resulting in an *embedded manifold*. Another option is to combine equivalent points in one element, resulting in a more abstract manifold called *quotient manifold*. In this thesis we consider the latter approach.

The choice comes from the invariance property (3.1). In order to solve (2.3), we are interested in subspaces rather than particular matrices. However, in numerical algorithms we work with matrices and there are infinitely many matrices representing the same subspace. Combining matrices that are equivalent in this sense, isolates the solutions of (2.3).

In this thesis, we consider two possible quotient manifolds, namely the Grassmann manifold and a manifold denoted by $\mathbb{R}_*^{n\times p}/O_p$. They are discussed in some more detail in Section 3.4 and Section 3.5. The latter manifold is used for our Newton methods described in the next two chapters. We look for zeros of the cost functions (4.1b) and (5.4), defined on (a product of three) $\mathbb{R}_*^{n\times p}$ ($\mathbb{R}_*^{n\times p}$ being the set of all full-rank $(n \times p)$-matrices) and having nonisolated zeros in the sense of (3.1). If we perform as little quotienting as possible in order to isolate the zeros of these functions, we obtain (a product of three) $\mathbb{R}_*^{n\times p}/O_p$. The Grassmann manifold on the other hand, is a well-studied manifold in the theory of matrix manifolds and removes more unused information from the cost function. It is used for our trust-region and conjugate gradient based algorithm in Chapter 6 and Chapter 7, respectively.

3.3 Quotient matrix manifolds

The following notation will be used throughout the thesis. The set

$$\mathbb{R}_*^{n\times p} = \{\mathbf{X} \in \mathbb{R}^{n\times p} : \det(\mathbf{X}^T\mathbf{X}) \neq 0\} \tag{3.2}$$

consists of all full-rank $(n \times p)$-matrices, $n \geq p$. The matrices

$$\mathrm{sym}(\mathbf{B}) = (\mathbf{B}+\mathbf{B}^T)/2 \tag{3.3}$$

and

$$\mathrm{skew}(\mathbf{B}) = (\mathbf{B}-\mathbf{B}^T)/2 \tag{3.4}$$

denote the terms of the decomposition of a square matrix $\mathbf{B}$ into a symmetric term and a skew-symmetric term.

Consider the equivalence relation $\sim$ on $\mathbb{R}_*^{n\times p}$, defined by

$$\mathbf{X} \sim \mathbf{X}_1 \quad \Longleftrightarrow \quad \text{there exists } \mathbf{Q} \in O_p, \text{ s.t. } \mathbf{X}_1 = \mathbf{X}\mathbf{Q},$$

where O_p is the orthogonal group (the set of all orthogonal $(p \times p)$-matrices). We define $[\mathbf{X}] = \{\mathbf{X}_1 : \mathbf{X} \sim \mathbf{X}_1\}$, i.e., $[\mathbf{X}]$ is the set of all matrices that can be obtained by multiplying $\mathbf{X}$ from the right by an orthogonal matrix. $[\mathbf{X}]$ is often also denoted as $\mathbf{X}O_p$. Any two sets $[\mathbf{X}]$ and $[\mathbf{Y}]$ are either disjoint or coincide and the union of all such sets equals $\mathbb{R}_*^{n\times p}$. For future purposes, we also mention that the union of all sets $[\mathbf{X}]$ with $\mathbf{X} \in St(p,n)$ equals $St(p,n)$. The sets $[\mathbf{X}]$ are called *equivalence classes* and the division of the original space ($\mathbb{R}_*^{n\times p}$ or $St(p,n)$) by subsets $[\mathbf{X}]$ is called *partition*. The *quotient sets* $\mathbb{R}_*^{n\times p}/O_p$ and $St(p,n)/O_p$ denote the sets of all equivalence classes. The sets $\mathbb{R}_*^{n\times p}$ and $St(p,n)$ are called *total spaces*.

In general, a quotient set can be very abstract. In this thesis we mainly use the two quotient sets mentioned above. For illustration purposes only, we present here a simpler example, a special case where $p = 1$. Consider $\mathbb{R}^n$ and let $\mathbb{R}_*^n$ denote $\mathbb{R}^n$ without its origin. The quotient $\mathbb{R}_*^n/\sim$, where $\sim$ is the equivalence relation

$$\mathbf{x} \sim \mathbf{x}_1 \quad \Longleftrightarrow \quad \text{there exists } a \in \mathbb{R}_*, \text{ s.t. } \mathbf{x}_1 = a\mathbf{x}$$

is called the *real projective space*, see also [5, §3.4.3]. Its elements are all directions in $\mathbb{R}^n$, i.e., all straight lines passing through the origin of $\mathbb{R}^n$. An illustration is given in Figure 3.2.

The quotient sets that we consider are a bit more complicated but are based on the same idea. In each equivalence class all elements have the same column space and in the case of $St(p,n)/O_p$ column spaces of different equivalence classes are different. There is a one-to-one correspondence between $St(p,n)/O_p$ and the Grassmann manifold (the set of all p-dimensional subspaces in $\mathbb{R}^n$).

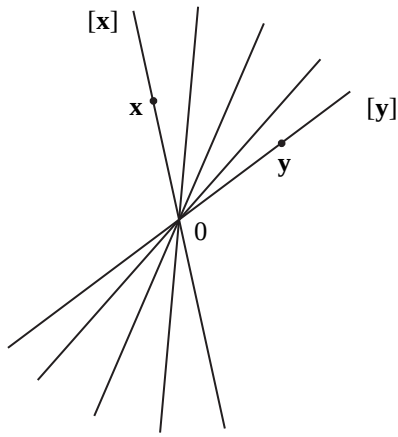


Figure 3.2: *Illustration of a quotient manifold.*

We now define some additional abstractions borrowed from the theory of fiber bundles [79], or more specifically from the theory of Riemannian submersions [107]. They will be necessary for the developments of the algorithms later on. Let $\overline{M}$ be an open subset of $\mathbb{R}^{n\times p}$ that is invariant by right multiplication by orthogonal matrices, for example $\mathbb{R}^{n\times p}_*$. $M = \overline{M}/O_p$ is then a quotient manifold [5]. If $\mathbf{X}O_p = \{\mathbf{XQ}, \mathbf{Q} \in O_p\} \in M$, the columns of $\mathbf{X}$ span the subspace corresponding to $\mathbf{X}O_p$ and $\mathbf{X}$ is a representation of $\mathbf{X}O_p$. We denote the tangent plane of M at $\mathbf{X}O_p$ by $T_{\mathbf{X}O_p}M$ and let $\xi \in T_{\mathbf{X}O_p}M$. For each $\mathbf{X}$, there are many different representations of ξ in the tangent plane $T_{\mathbf{X}}\overline{M}$ and this leads to ambiguities. For each tangent vector of $T_{\mathbf{X}O_p}M$, a unique representation in $T_{\mathbf{X}}\overline{M}$ can be assigned in the following way. Let $\mathscr{V}_{\mathbf{X}}$ be the tangent space of $\mathbf{X}O_p$, i.e.,

$$\mathscr{V}_{\mathbf{X}} = T_{\mathbf{X}}\mathbf{X}O_p\,.$$

$\mathscr{V}_{\mathbf{X}}$ is called the *vertical space* of M at $\mathbf{X}$. A subspace of $T_{\mathbf{X}}\overline{M}$ that is complementary to the vertical space is the *horizontal space*, denoted by $\mathscr{H}_{\mathbf{X}}$. In the horizontal space, there exists only one valid representation $\overline{\xi}$ of ξ called the *horizontal lift* of ξ at $\mathbf{X}$. This is the one that should be used when the theoretical algorithms are implemented as a computer program. The horizontal lift $\overline{\xi}$ is normally obtained by projecting a representation of ξ at $\mathbf{X}$ onto the horizontal space $\mathscr{H}_{\mathbf{X}}$. For more details we refer to [5, §3.5.8].

3.4 Grassmann manifold

The *Grassmann manifold* is the set of all p-dimensional subspaces in $\mathbb{R}^n$. It can be thought of as the quotient manifold $M = St(p,n)/O_p$. For the Grassmann manifold we also refer to [48, 4].

The vertical space at $\mathbf{X} \in St(p,n)$ is defined to be the tangent space at $\mathbf{X}$ to the equivalence class $\mathbf{X}O_p$, i.e.,

$$\mathscr{V}_{\mathbf{X}} = T_{\mathbf{X}}\mathbf{X}O_p = \{\mathbf{X}\mathbf{M} : \mathbf{M} = -\mathbf{M}^T \in \mathbb{R}^{p\times p}\}.$$

The tangent space to $St(p,n)$ at $\mathbf{X}$ is

$$\begin{aligned} T_{\mathbf{X}}St(p,n) &= \{\mathbf{Z} : \mathbf{X}^T\mathbf{Z} + \mathbf{Z}^T\mathbf{X} = \mathbf{0}\} \\ &= \{\mathbf{X}\mathbf{M} + \mathbf{X}_\perp\mathbf{K} : \mathbf{M} = -\mathbf{M}^T \in \mathbb{R}^{p\times p}, \mathbf{K} \in \mathbb{R}^{(n-p)\times p}\}, \end{aligned}$$

where $\mathbf{X}_\perp$ is any matrix such that $\begin{bmatrix}\mathbf{X} & \mathbf{X}_\perp\end{bmatrix} \in O_n$. It is obtained by differentiating the equation $\mathbf{X}^T\mathbf{X} = \mathbf{I}$, which determines the matrices in $St(p,n)$. Let $\mathbf{Z}_1, \mathbf{Z}_2 \in T_{\mathbf{X}}St(p,n)$. We define an inner product on $T_{\mathbf{X}}St(p,n)$ in the following way

$$\langle \mathbf{Z}_1, \mathbf{Z}_2 \rangle_{\mathbf{X}} = \mathrm{trace}(\mathbf{Z}_1^T\mathbf{Z}_2),$$

where trace($\mathbf{B}$) is the sum of the diagonal elements of the matrix $\mathbf{B}$. We choose the horizontal space at $\mathbf{X}$ to be the orthogonal complement of the vertical space in $T_{\mathbf{X}}St(p,n)$,

$$\mathscr{H}_{\mathbf{X}} = \{\mathbf{X}_\perp\mathbf{K} : \mathbf{K} \in \mathbb{R}^{(n-p)\times p}\}.$$

Given $\mathbf{Z}_{\mathbf{X}O_p} \in T_{\mathbf{X}O_p}M$, there exists a unique horizontal lift $\overline{\mathbf{Z}}_{\mathbf{X}} \in \mathscr{H}_{\mathbf{X}}$ of $\mathbf{Z}_{\mathbf{X}O_p}$ at $\mathbf{X}$, such that $\mathrm{D}\pi(\mathbf{X})[\overline{\mathbf{Z}}_{\mathbf{X}}] = \mathbf{Z}_{\mathbf{X}O_p}$, where

$$\begin{array}{rccc} \pi: & St(p,n) & \to & M \\ & \mathbf{X} & \mapsto & \mathbf{X}O_p \end{array}$$

is the *quotient map*. It can be shown that $\overline{\mathbf{Z}}_{\mathbf{X}\mathbf{Q}} = \overline{\mathbf{Z}}_{\mathbf{X}}\mathbf{Q}$ for all $\mathbf{Q} \in O_p$. Observe that $\langle \overline{\mathbf{Z}}^{(1)}_{\mathbf{X}\mathbf{Q}}, \overline{\mathbf{Z}}^{(2)}_{\mathbf{X}\mathbf{Q}} \rangle = \langle \overline{\mathbf{Z}}^{(1)}_{\mathbf{X}}, \overline{\mathbf{Z}}^{(2)}_{\mathbf{X}} \rangle$, hence $\langle\langle \mathbf{Z}^{(1)}_{\mathbf{X}O_p}, \mathbf{Z}^{(2)}_{\mathbf{X}O_p} \rangle\rangle = \langle \overline{\mathbf{Z}}^{(1)}_{\mathbf{X}}, \overline{\mathbf{Z}}^{(2)}_{\mathbf{X}} \rangle$ is a well-defined inner product on $T_{\mathbf{X}O_p}M$. Finally, the *orthogonal projection* onto the horizontal space at $\mathbf{X}$ is given by

$$P^h_{\mathbf{X}}(\mathbf{Z}) = (\mathbf{I} - \mathbf{X}\mathbf{X}^T)\mathbf{Z}. \tag{3.5}$$

The elements of the vertical space can be thought of as variations that do not modify the column space of $\mathbf{X}$, whereas the elements of the horizontal space, applied to $\mathbf{X}$, modify the span of $\mathbf{X}$. The projection (3.5) reflects the fact that only variations of $\mathbf{X}$ that are orthogonal to the column space of $\mathbf{X}$ are important. Any other variations are irrelevant, since they are variations in the same equivalence class so that they do not affect points in $St(p,n)/O_p$.

From now on, while working on a Grassmann manifold, we abuse notation and often use g for $\overline{g}$, $\mathbf{X}$ for $\mathbf{X}O_p$, $\mathbf{Z}_{\mathbf{X}}$ for $\overline{\mathbf{Z}}_{\mathbf{X}}$ and $\langle\cdot,\cdot\rangle$ for $\langle\langle\cdot,\cdot\rangle\rangle$. This is acceptable since $g(\mathbf{X}O_p) = \overline{g}(\mathbf{X})$ and $\langle\langle \mathbf{Z}^{(1)}_{\mathbf{X}O_p}, \mathbf{Z}^{(2)}_{\mathbf{X}O_p} \rangle\rangle = \langle \overline{\mathbf{Z}}^{(1)}_{\mathbf{X}}, \overline{\mathbf{Z}}^{(2)}_{\mathbf{X}} \rangle$.

3.5 The manifold $\mathbb{R}_*^{n\times p}/O_p$

In this section we discuss the manifold $\mathbb{R}_*^{n\times p}/O_p$. A short description of the manifold can be found in [3], see also [76].

Let

$$\begin{array}{rccc} \pi: & \mathbb{R}_*^{n\times p} & \to & \mathbb{R}_*^{n\times p}/O_p \\ & \mathbf{X} & \mapsto & \mathbf{X}O_p \end{array} \tag{3.6}$$

be the quotient map. As mentioned earlier, a point $\pi(\mathbf{X}) \in \mathbb{R}_*^{n\times p}/O_p$ can be (numerically) represented by any element of its equivalence class $\mathbf{X}O_p = \pi^{-1}(\pi(\mathbf{X}))$.

The set $\mathbb{R}_*^{n\times p}$ is an open subset of $\mathbb{R}^{n\times p}$. It is thus an open submanifold of the linear manifold $\mathbb{R}^{n\times p}$. It can be shown that the map

$$\psi : O_n \times \mathbb{R}_*^{n\times p} \to \mathbb{R}_*^{n\times p} : (\mathbf{Q}, \mathbf{X}) \mapsto \mathbf{X}\mathbf{Q}$$

is a free and proper Lie group action on the manifold $\mathbb{R}_*^{n\times p}$. By the quotient manifold theorem (see, e.g., [87, Th. 9.16]), the orbit space $\mathbb{R}_*^{n\times p}/O_p$ is a quotient manifold, i.e., the set $\mathbb{R}_*^{n\times p}/O_p$ is turned into a manifold by endowing it with the unique differentiable structure that makes the quotient map π a submersion. As a consequence, each equivalence class $\mathbf{X}O_p = [\mathbf{X}] = \pi^{-1}(\pi(\mathbf{X}))$, $\mathbf{X} \in \mathbb{R}_*^{n\times p}$ is an embedded submanifold of $\mathbb{R}_*^{n\times p}$.

The vertical space at $\mathbf{X} \in \mathbb{R}_*^{n\times p}$ is the tangent space at $\mathbf{X}$ to the equivalent class $\mathbf{X}O_p$. It is given by the following formula

$$\mathscr{V}_\mathbf{X} = T_\mathbf{X}\mathbf{X}O_p = \mathbf{X}\, T_\mathbf{I}O_p = \{\mathbf{X}\mathbf{M} : \mathbf{M} = -\mathbf{M}^T \in \mathbb{R}^{p\times p}\}.$$

The horizontal space $\mathscr{H}_\mathbf{X}$ has to be such that its direct sum with $\mathscr{V}_\mathbf{X}$ yields $\mathbb{R}^{n\times p}$, since $T_\mathbf{X}\mathbb{R}_*^{n\times p} = \mathbb{R}^{n\times p}$. We can take

$$\mathscr{H}_\mathbf{X} = \{\mathbf{X}\mathbf{S} + \mathbf{X}^\perp\mathbf{K} : \mathbf{S} = \mathbf{S}^T \in \mathbb{R}^{p\times p}, \mathbf{K} \in \mathbb{R}^{(n-p)\times p}, \mathbf{X}^\perp \in \mathbb{R}_*^{n\times(n-p)}, \mathbf{X}^T\mathbf{X}^\perp = \mathbf{0}\}. \tag{3.7}$$

Finally, the projection onto $\mathscr{H}_\mathbf{X}$ along $\mathscr{V}_\mathbf{X}$ is given by

$$P_\mathbf{X}^h(\mathbf{Z}) \quad = \quad \mathbf{Z} - \mathbf{X}\,\mathrm{skew}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Z}), \tag{3.8}$$

where skew($\mathbf{B}$) is defined as in (3.4).

We mention that the horizontal space $\mathscr{H}_\mathbf{X}$ is the orthogonal complement of the vertical space $\mathscr{V}_\mathbf{X}$ with respect to the inner product $\langle\cdot,\cdot\rangle$ defined by

$$\langle \mathbf{Z}_1, \mathbf{Z}_2\rangle_\mathbf{X} = \mathrm{trace}(\mathbf{Z}_1^T P_\mathbf{X}^p(\mathbf{Z}_2) + \mathbf{Z}_1^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-2}\mathbf{X}^T\mathbf{Z}_2), \tag{3.9}$$

where $P_\mathbf{X}^p(\mathbf{Z}) = (\mathbf{I} - \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T)\mathbf{Z}$. Indeed, $\{\mathbf{Z} \in \mathbb{R}^{n\times p} : \langle \mathbf{W}, \mathbf{Z}\rangle = 0 \text{ for all } \mathbf{W} \in \mathscr{V}_\mathbf{X}\} = \{\mathbf{X}\mathbf{S} + \mathbf{X}_\perp\mathbf{K} : \mathbf{S}^T = \mathbf{S}\}$. The canonical Riemannian metric on $\mathbb{R}_*^{n\times p}$ yields a more complex

formula for the horizontal space and for the projection onto it. We prefer to consider a more complicated formula for the metric and simple formulas for the horizontal space and the projection $P^h_{\mathbf{X}}$ that will be used for derivations.

For a function $F : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p}$ with a symmetry property

$$F(\mathbf{XQ}) = F(\mathbf{X})\mathbf{Q}, \text{ for all } \mathbf{Q} \in O_p,$$

it can be checked that the horizontal vector field $\overline{\xi}$, defined by

$$\overline{\xi}_{\mathbf{X}} := P^{\mathrm{h}}_{\mathbf{X}}(F(\mathbf{X}))$$

satisfies $\mathrm{D}\pi(\mathbf{X})[\overline{\xi}_{\mathbf{X}}] = \mathrm{D}\pi(\mathbf{XQ})[\overline{\xi}_{\mathbf{XQ}}]$ for all $\mathbf{Q} \in O_p$. Thus, $\overline{\xi}$ is the horizontal lift of a vector field ξ on the quotient manifold $\mathbb{R}^{n\times p}_*/O_p$.

3.6 Retraction and vector transport

In this section, we define two notions that help generalizing optimization procedures, such as line search, to manifolds. A function named retraction has to be defined in order to move in the direction of a tangent vector but land on a manifold. In $\mathbb{R}^n$, this concept is straightforward. Adding a direction vector to the current point, i.e., to the vector representing the current point, gives again a point in $\mathbb{R}^n$. For manifolds, some more effort has to be done. Simply adding a tangent vector to a point on a manifold would in general give a point that does not belong to the manifold. The idea of retraction is explained in more detail in Section 3.6.1.

In Section 3.6.2, we discuss the so-called vector transport. Such function is necessary when working on two tangent spaces at the same time, for example in formulas comparing two consecutive gradient vectors. The vector transport is a mapping that transports a tangent vector from one tangent plane to another tangent plane.

3.6.1 Retraction

A *retraction* R defines a unique point on a manifold corresponding to a given tangent vector. This concept was introduced in [119] and further investigated in [9]. The retraction at a point $\mathbf{X}$ on a manifold M is denoted by $R_{\mathbf{X}}$. It is a smooth function $R_{\mathbf{X}} : T_{\mathbf{X}}M \to M$ sending the zero of $T_{\mathbf{X}}M$ to $\mathbf{X}$ itself and such that its derivative is the identity map on $T_{\mathbf{X}}M$. The latter condition can also be interpreted as follows. For any $\xi \in T_{\mathbf{X}}M$ and its corresponding curve $\gamma_\xi : t \mapsto R_{\mathbf{X}}(t\xi)$, it holds $\dot{\gamma}_\xi(0) = \xi$. For a formal definition of a retraction we refer to [5, Def. 4.1.1]. The concept of retraction is illustrated in Figure 3.3.

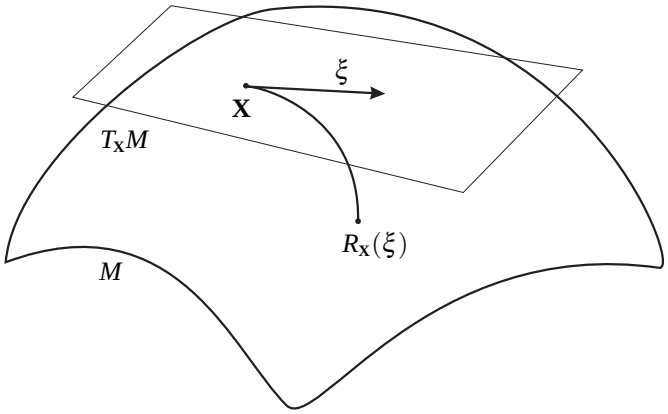


Figure 3.3: *Retraction.*

For derivations of practical algorithms, an important property of a retraction is its ability to translate a cost function $f : M \to \mathbb{R}$ locally to the tangent plane of the current point $\mathbf{X}$. The new function $\hat{f} : TM \to \mathbb{R}$, where $TM = \bigcup_{\mathbf{X} \in M} T_{\mathbf{X}}M$, is defined at $\mathbf{X}$ as follows

$$\hat{f}_{\mathbf{X}} = f \circ R_{\mathbf{X}}$$

and is termed *pullback* of f through R. The advantage of the pullback $\hat{f}_{\mathbf{X}}$ over the original f is that $\hat{f}_{\mathbf{X}}$ is a function defined on a vector space.

The choice of retraction is important. The first obvious choice is the exponential map. However, depending on the manifold, this choice may be computationally inefficient [93]. There are often much better functions in the sense that they satisfy all necessary conditions in order to be a retraction and they are less expensive. A retraction can be thought of as a cheap approximation of the exponential map, without destroying the convergence behavior of the optimization methods.

For the Grassmann manifold, or its equivalent $M = St(p,n)/O_p$, a natural choice of a retraction is the following one [5, §4.1.2]

$$R_{\mathbf{X}O_p}(\mathbf{Z}) = \mathrm{qf}(\mathbf{X} + \overline{\mathbf{Z}})O_p\,, \tag{3.10}$$

where qf denotes the **Q** factor of the thin **QR** decomposition [57, §5.2] and $\mathbf{Z}$ is a tangent vector. This choice is also motivated by the fact that we are only interested in column spaces of the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ from (2.3) and not in their actual values. For the manifold $\mathbb{R}_*^{n\times p}/O_p$, an even simpler retraction function could be taken, namely

$$R_{\mathbf{X}O_p}(\mathbf{Z}) = (\mathbf{X} + \overline{\mathbf{Z}})O_p\,. \tag{3.11}$$

3.6.2 Vector transport

For our conjugate gradient algorithm in Chapter 7, we need to compare gradient vectors from two different tangent planes. For this purpose, one of the vectors has to be moved to the tangent space of the other one. This can be done by a parallel translation, or by a (well-chosen) vector transport. The vector transport has a different purpose than a retraction but is a similar concept in the sense that it is a cheap version of parallel translation, being just as useful as the parallel translation at the same time.

A *vector transport*, denoted by $\mathscr{T}$, is a collection of mappings at each $\mathbf{X} \in M$, such that at a given $\mathbf{X}$,

$$\begin{array}{rccl} \mathscr{T}: & T_{\mathbf{X}}M \times T_{\mathbf{X}}M & \to & TM, \\ & (\eta_{\mathbf{X}}, \xi_{\mathbf{X}}) & \mapsto & \mathscr{T}_{\eta_{\mathbf{X}}}\xi_{\mathbf{X}} \end{array}$$

satisfies an associativity, a consistency and a linearity condition. This means that there exists a retraction R, such that the initial point of $\mathscr{T}_{\eta_{\mathbf{X}}}\xi_{\mathbf{X}}$ is at point $R_{\eta_{\mathbf{X}}}$ (associativity). Further, the vector transport for $\eta_{\mathbf{X}} = 0_{\mathbf{X}}$ is the identity on $T_{\mathbf{X}}M$ (consistency) and the vector transport of a linear combination of two vectors is the linear combination of the transported vectors (linearity in $\xi_{\mathbf{X}}$). We refer to [5, Def. 8.1.1] for the precise formulation. The concept is illustrated in Figure 3.4.

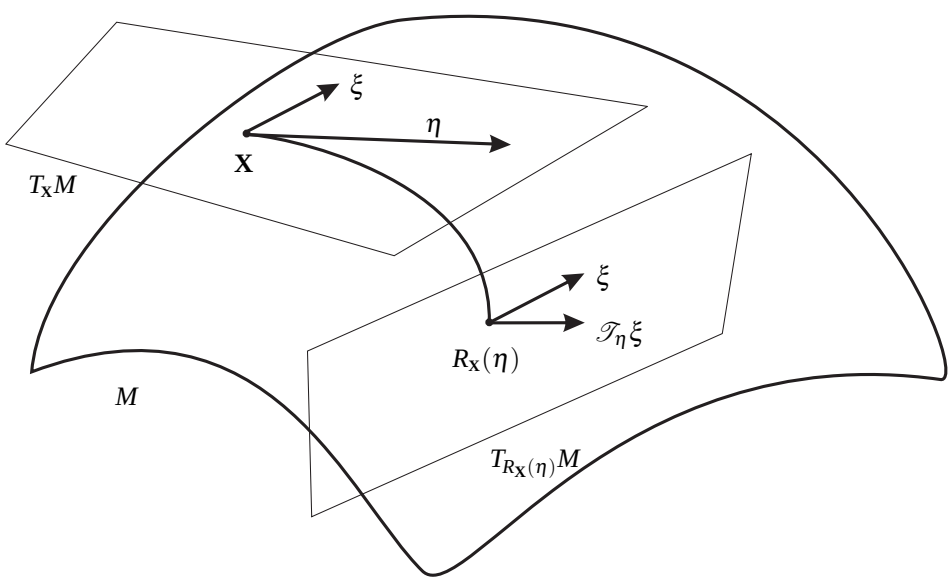


Figure 3.4: *Vector transport.*

As in the case of retraction, there are many possibilities for a vector transport in a given setting. In this thesis, we will need the concept of vector transport only for a conjugate gradient algorithm, where we choose to work on the Grassmann manifold. A simple vector transport in this case is

$$\overline{(\mathscr{T}_{\eta_{\mathbf{X}}}\xi_{\mathbf{X}})}_{\mathrm{qf}(\mathbf{X}+\overline{\eta}_{\mathbf{X}})} = P^h_{\mathrm{qf}(\mathbf{X}+\overline{\eta}_{\mathbf{X}})}\overline{\xi}_{\mathbf{X}}, \tag{3.12}$$

where $\eta_{\mathbf{X}}$ and $\xi_{\mathbf{X}}$ are two tangent vectors at point $\mathbf{X}O_p$ and $\overline{\xi}_{\mathbf{X}}$ and $\overline{\eta}_{\mathbf{X}}$ are the horizontal lifts at $\mathbf{X}$ of $\xi_{\mathbf{X}}$ and $\eta_{\mathbf{X}}$ respectively. $P^h_{\mathbf{Y}}$ is the orthogonal projection (3.5) onto the horizontal space of the point $\mathbf{Y}$. Note that $\mathrm{qf}(\mathbf{X}+\overline{\eta}_{\mathbf{X}})O_p = R_{\mathbf{X}O_p}\eta_x$.

3.7 Conclusions

In this chapter, we presented an invariance property of the problem (2.3) which led to considering working on a manifold in order to remove redundant information. We introduced some basic concepts of optimization on manifolds. In particular, we are interested in quotient matrix manifolds. We gave a feeling of what a quotient manifold is and listed some of the basic technical definitions, allowing developing algorithms on a quotient manifold.

In particular, we have considered two quotient manifolds, namely the Grassmann manifold, represented by $St(p,n)/O_p$, and the manifold $\mathbb{R}^{n\times p}_*/O_p$. Both manifolds have their advantages and will be used in the algorithms in the next chapters. Some technical formulas for these manifolds were presented. Finally, we have introduced the concepts of retraction and vector transport. These two types of functions replace their more expensive counterparts, the exponential map and the parallel transport respectively. Particular choices for a retraction and a vector transport were presented for the two quotient manifolds that are considered in the thesis.

Part II

Algorithms

Chapter 4

A geometric Newton method for Oja's vector field

Newton's method for solving the matrix equation $F(\mathbf{X}) \equiv \mathbf{AX} - \mathbf{XX}^T\mathbf{AX} = \mathbf{0}$ runs up against the fact that its zeros are not isolated. This is due to a symmetry of F by the action of the orthogonal group. We show how differential-geometric techniques can be exploited to remove this symmetry and obtain a "geometric" Newton algorithm that finds the zeros of F. The geometric Newton method does not suffer from the degeneracy issue that stands in the way of the original Newton method.

The problem discussed in this chapter involves matrices and not higher-order tensors, which are of prime interest to this thesis. However, starting from the technique proposed in the current chapter, we were able to develop a differential-geometric Newton method for computing the best multilinear rank approximation of higher-order tensors, as presented in the next chapter. The material in the current chapter has been published in [3]. The main part of it is written by its first author.

4.1 Introduction

Oja's flow [103, 104], in its averaged version

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{X}(t) = F(\mathbf{X}(t)), \tag{4.1a}$$

where F denotes the vector field

$$F : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : \mathbf{X} \mapsto \mathbf{AX} - \mathbf{XX}^T\mathbf{AX}, \tag{4.1b}$$

$\mathbf{A}$ is a symmetric positive-definite $n \times n$ matrix and $p < n$, is well known for its principal component analysis properties. For all initial conditions $\mathbf{X}_0$, the ordinary differential equation (4.1) has a unique solution curve $t \mapsto \mathbf{X}(t)$ on the interval $[0,\infty)$ [143, Th. 2.1], the limit $\mathbf{X}(\infty) = \lim_{t\to\infty} \mathbf{X}(t)$ exists, the convergence to $\mathbf{X}(\infty)$ is exponential, and $\mathbf{X}(\infty)$ is a zero of Oja's vector field F (4.1b) [143, Th. 3.1].

Observe that the zeros of F are the solutions $\mathbf{X} \in \mathbb{R}^{n\times p}$ of the matrix equation

$$\mathbf{AX} = \mathbf{X}\mathbf{X}^T\mathbf{AX},$$

which implies that the columns of $\mathbf{AX}$ are linear combinations of the columns of $\mathbf{X}$. Letting

$$\mathrm{col}(\mathbf{Y}) = \{\mathbf{Ya} : \mathbf{a} \in \mathbb{R}^p\}$$

denote the column space of $\mathbf{Y} \in \mathbb{R}^{n\times p}$, it follows that all zeros $\mathbf{X}$ of (4.1b) satisfy $\mathbf{A}\,\mathrm{col}(\mathbf{X}) \subseteq \mathrm{col}(\mathbf{X})$, i.e., $\mathrm{col}(\mathbf{X})$ is an invariant subspace of $\mathbf{A}$. Moreover, $\mathbf{X}(\infty)$ is an orthonormal matrix ($\mathbf{X}^T(\infty)\mathbf{X}(\infty) = \mathbf{I}_p$), thus of full rank, whenever the initial condition $\mathbf{X}(0)$ is of full rank [143, Prop. 3.1]. Assuming that $\mathbf{X}(0)$ has full rank, it holds that $\mathrm{col}(\mathbf{X}(\infty))$ is the p-dimensional principal subspace of $\mathbf{A}$ (assumed to be unique) if and only if $\mathrm{rank}(\mathbf{X}_0^T\mathbf{V}_p) = p$, where $\mathbf{V}_p$ is any $(n \times p)$-matrix whose column space is the p-dimensional principal subspace of $\mathbf{A}$ [143, Th. 5.1]. This condition means that $\mathrm{col}(\mathbf{X}(0))$ does not contain any direction orthogonal to the p-dimensional principal subspace of $\mathbf{A}$. The set of all initial conditions that do not satisfy this condition is a negligible set, i.e., Oja's flow asymptotically computes the p-dimensional principal subspace of $\mathbf{A}$ for almost all initial conditions. We also point out that Oja's flow induces a subspace flow, called the power flow [6].

Because of these remarkable properties, Oja's flow has been and remains the subject of much attention, in its form (4.1) as well as in several modified versions; see, for example, [143, 142, 46, 94, 74] . However, turning Oja's flow into a principal component algorithm implementable in a digital computer requires to discretize the flow in such a way that its good convergence properties are preserved. Since the solutions $\mathbf{X}(t)$ converge exponentially to their limit point $\mathbf{X}(\infty)$, it follows that the sequence of equally spaced discrete-time samples $(\mathbf{X}(kT))_{k\in\mathbb{N}}$ converges only $\mathbf{Q}$-linearly [108] to $\mathbf{X}(\infty)$. Therefore, numerical integration methods that try to compute accurately the solution trajectories of Oja's flow are not expected to converge faster than linearly.

Nevertheless, if the ultimate goal is to compute the principal eigenspace of $\mathbf{A}$, then it is tempting to try to accelerate the convergence when the iterates get close to the limit point $\mathbf{X}(\infty)$, using techniques akin to those proposed in [64, 53, 77, 90], in order to obtain a superlinear algorithm. To this end, it is interesting to investigate how superlinearly convergent methods perform for finding a zero of Oja's vector field (4.1b). Since Newton's method can be thought of as the prototype superlinear algorithm to which all other superlinear algorithms relate, we propose to investigate the behavior of Newton's method applied to the problem of finding a zero of Oja's vector field (4.1b).

A crucial hypothesis in the classical local convergence analysis of Newton's method (see, e.g., [47]) is that the targeted zero is nondegenerate. As it turns out, the zeros of Oja's vector field F (4.1b) are never isolated, because F displays a property of symmetry under the action of the orthogonal group O_p on the set $\mathbb{R}^{n\times p}$. Therefore, the classical superlinear convergence result of Newton's method is void in the case of F, and indeed our numerical experiments show that Newton's method on $\mathbb{R}^{n\times p}$ for F performs poorly (see Figure 4.1).

A way to address this difficulty is to modify the vector field F so as to break the symmetry while preserving a useful relation between the zeros of F and the invariant subspaces of $\mathbf{A}$. For example, Oja *et al.* [105, 106] and Xu [141] consider $F : \mathbf{X} \mapsto \mathbf{AXD} - \mathbf{XDX}^T\mathbf{AX}$, where $\mathbf{D}$ is a nondegenerate positive $(p \times p)$ diagonal matrix. They prove that p principal eigenvectors (not merely the subspace spanned by them) are derived, and hence no indefiniteness occurs provided that there is no degeneracy in the eigenvalues of $\mathbf{A}$ (i.e., $\mathbf{A}$ has n distinct eigenvalues). A detailed analysis appears in [144, 145]. The algorithm is further extended to derive the minor subspace in a similar form by Chen and Amari [23], where stability of the algorithm is studied in the geometrical framework.

In this chapter, rather than modifying F (4.1b), we propose a remedy to the degeneracy that consists in "quotienting out" the symmetry of F. Conceptually speaking, instead of performing a Newton iteration in $\mathbb{R}^{n\times p}$, we perform a Newton iteration on the quotient space $\mathbb{R}_*^{n\times p}/O_p$ (defined in Chapter 3). We exploit the Riemannian quotient manifold structure of $\mathbb{R}_*^{n\times p}/O_p$ to formulate a Newton method on this set, following the framework developed in [5]. The resulting Newton equation is a linear matrix equation that can be solved by various numerical approaches. It follows from the theory of Newton method on manifolds (see, e.g., [9, 5]), and from a careful analysis of the zeros of the vector field, that the obtained algorithm converges locally superlinearly to the set of orthonormal bases of invariant subspaces of $\mathbf{A}$. Our numerical experiments show that the method behaves as expected.

4.2 Plain Newton method for Oja's vector field

We assume throughout the chapter that $\mathbf{A}$ is a real symmetric positive-definite $n \times n$ matrix. For simplicity, we also assume that the eigenvalues of $\mathbf{A}$ satisfy

$$\lambda_1 > \cdots > \lambda_n, \tag{4.2}$$

i.e., all the eigenvalues of $\mathbf{A}$ are simple. Hence the p-dimensional invariant subspaces of $\mathbf{A}$ are isolated.

Newton's method in $\mathbb{R}^{n\times p}$ for Oja's vector field F (4.1b) consists of iterating the map $\mathbf{X} \mapsto \mathbf{X}_+$ defined by solving

$$\mathrm{D}F(\mathbf{X})[\mathbf{Z}] \equiv \mathbf{A}\mathbf{Z} - \mathbf{Z}\mathbf{X}^T\mathbf{A}\mathbf{X} - \mathbf{X}\mathbf{Z}^T\mathbf{A}\mathbf{X} - \mathbf{X}\mathbf{X}^T\mathbf{A}\mathbf{Z} = -F(\mathbf{X}) \tag{4.3}$$

$$\mathbf{X}_+ = \mathbf{X} + \mathbf{Z}. \tag{4.4}$$

The following proposition is a well-known characterization of the zeros of F.

Proposition 4.2.1 (zeros of F (4.1b)) *Let $\mathbf{X} \in \mathbb{R}^{n\times p}$ be of full rank. Then the following two conditions are equivalent:*

1. $F(\mathbf{X}) = \mathbf{0}$, *i.e.,*

$$\mathbf{A}\mathbf{X} = \mathbf{X}\mathbf{X}^T\mathbf{A}\mathbf{X}. \tag{4.5}$$

2. $\mathrm{col}(\mathbf{X})$ *is an invariant subspace of* $\mathbf{A}$ *and* $\mathbf{X}$ *is orthonormal (*$\mathbf{X}^T\mathbf{X} = \mathbf{I}$*).*

Proof $1 \Rightarrow 2$. We have that $\mathbf{A}\mathbf{X} = \mathbf{X}(\mathbf{X}^T\mathbf{A}\mathbf{X})$, thus $\mathbf{A}\,\mathrm{col}(\mathbf{X}) \subseteq \mathrm{col}(\mathbf{X})$. (More precisely, since $\mathbf{X}$ has full rank and $\mathbf{A}$ is positive-definite, it follows that $\mathbf{X}^T\mathbf{A}\mathbf{X}$ is invertible and thus $\mathbf{A}\,\mathrm{col}(\mathbf{X}) = \mathrm{col}(\mathbf{X})$.) Moreover, multiplying (4.5) by $\mathbf{X}^T$ on the left yields $\mathbf{X}^T\mathbf{A}\mathbf{X} = \mathbf{X}^T\mathbf{X}\mathbf{X}^T\mathbf{A}\mathbf{X}$, which implies that $\mathbf{X}^T\mathbf{X} = \mathbf{I}$ since $\mathbf{X}^T\mathbf{A}\mathbf{X}$ is invertible.

$2 \Rightarrow 1$. Since $\mathrm{col}(\mathbf{X})$ is an invariant subspace of $\mathbf{A}$, there is a matrix $\mathbf{M}$ such that $\mathbf{A}\mathbf{X} = \mathbf{X}\mathbf{M}$. Multiply this equation on the left by $\mathbf{X}^T$ to obtain that $\mathbf{M} = \mathbf{X}^T\mathbf{A}\mathbf{X}$ and thus (4.5). □

Hence, the set of all full-rank zeros of F is the finite union of the compact sets

$$\mathscr{S}_i := \{\mathbf{X} \in \mathbb{R}^{n\times p} : \mathrm{col}(\mathbf{X}) = \mathscr{E}_i, \mathbf{X}^T\mathbf{X} = \mathbf{I}\} \tag{4.6}$$

where $\mathscr{E}_1, \ldots, \mathscr{E}_N$ are the p-dimensional invariant subspaces of $\mathbf{A}$. (Note that N is finite; it is equal to $\binom{n}{p}$.) It is readily checked that $\mathscr{S}_i = \mathbf{V}_i O_p$, where O_p is the orthogonal group and $\mathbf{V}_i$ is an element of $\mathscr{S}_i$. It follows that the zeros of F are not isolated. Hence the Jacobian of F at the zeros of F is singular (i.e., the zeros are *degenerate*), and consequently, the classical result (see [47, Th. 5.2.1]) of local superlinear convergence of Newton's iteration to the nondegenerate zeros of F is void. This does not imply that Newton's method will fail, but there is a suspicion that it will behave poorly, and indeed, in Section 4.5 (see Figure 4.1), we report on numerical experiments showing that this is the case.

4.3 Newton's method on $\mathbb{R}_*^{n\times p}/O_p$

The degeneracy of the zeros of F is due to the following fundamental symmetry property:

$$F(\mathbf{XQ}) = F(\mathbf{X})\mathbf{Q}, \text{ for all } \mathbf{Q} \in O_p. \tag{4.7}$$

In this section, we propose a *geometric* Newton method that performs well with functions F that satisfy (4.7). This geometric Newton method evolves on the quotient space $\mathbb{R}_*^{n\times p}/O_p$, where this symmetry is removed. Then, in Section 4.4, we will return to the specific case where F is Oja's vector field (4.1b) and obtain a concrete numerical algorithm.

The general idea for the geometric Newton method is first to define a vector field ξ on the manifold $\mathbb{R}_*^{n\times p}/O_p$, whose zeros relate to those of F. The vector field ξ is specified in terms of its horizontal lift in $\mathbb{R}_*^{n\times p}$, see Section 3.5.

For $\mathbf{X} \in \mathbb{R}_*^{n\times p}$, $\mathbb{R}_*^{n\times p}$ as in (3.2), we define

$$P_{\mathbf{X}}^{\mathrm{p}} : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : \mathbf{Z} \mapsto P_{\mathbf{X}}^{\mathrm{p}}(\mathbf{Z}) = (\mathbf{I} - \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T)\mathbf{Z} \tag{4.8}$$

$$P_{\mathbf{X}}^{\mathrm{s}} : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : \mathbf{Z} \mapsto P_{\mathbf{X}}^{\mathrm{s}}(\mathbf{Z}) = \mathbf{X}\mathrm{sym}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Z}) \tag{4.9}$$

$$P_{\mathbf{X}}^{\mathrm{a}} : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : \mathbf{Z} \mapsto P_{\mathbf{X}}^{\mathrm{a}}(\mathbf{Z}) = \mathbf{X}\mathrm{skew}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Z}) \tag{4.10}$$

$$P_{\mathbf{X}}^{\mathrm{h}} = P_{\mathbf{X}}^{\mathrm{p}} + P_{\mathbf{X}}^{\mathrm{s}} : \mathbf{Z} \mapsto \mathbf{Z} - \mathbf{X}\mathrm{skew}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Z}), \tag{4.11}$$

where $\mathrm{sym}(\mathbf{B})$ (3.3) and $\mathrm{skew}(\mathbf{B})$ (3.4) denote the terms of the decomposition of a square matrix $\mathbf{B}$ into a symmetric term and a skew-symmetric term. We have $\mathbf{Z} = P_{\mathbf{X}}^{\mathrm{p}}(\mathbf{Z}) + P_{\mathbf{X}}^{\mathrm{s}}(\mathbf{Z}) + P_{\mathbf{X}}^{\mathrm{a}}(\mathbf{Z})$ for all $\mathbf{Z} \in \mathbb{R}^{n\times p}$. Observe that $\mathrm{im}(P_{\mathbf{X}}^{\mathrm{p}}) = \{\mathbf{Z} \in \mathbb{R}^{n\times p} : \mathbf{X}^T\mathbf{Z} = \mathbf{0}\}$, $\mathrm{im}(P_{\mathbf{X}}^{\mathrm{s}}) = \mathbf{X}\mathscr{S}_{\mathrm{sym}}(p)$, $\mathrm{im}(P_{\mathbf{X}}^{\mathrm{a}}) = \mathbf{X}\mathscr{S}_{\mathrm{skew}}(p)$, where

$$\mathscr{S}_{\mathrm{sym}}(p) = \{\mathbf{S} \in \mathbb{R}^{p\times p} : \mathbf{S}^T = \mathbf{S}\}$$

denotes the set of all symmetric matrices of order p and

$$\mathscr{S}_{\mathrm{skew}}(p) = \{\Omega \in \mathbb{R}^{p\times p} : \Omega^T = -\Omega\}$$

is the set of all *skew-symmetric* (or *antisymmetric*) matrices of order p. (The letter "p" stands for "perpendicular", "s" for symmetric, "a" for antisymmetric, and P^{h} is the projection onto the horizontal space, already defined as (3.8)).

Recall the quotient set $\mathbb{R}_*^{n\times p}/O_p$ from Section 3.5. Observe that

$$\mathscr{V}_{\mathbf{X}} = \mathrm{im}(P_{\mathbf{X}}^{\mathrm{a}})$$

and

$$\mathscr{H}_{\mathbf{X}} = \mathrm{im}(P_{\mathbf{X}}^{\mathrm{h}}).$$

For finding a zero of a vector field ξ on an abstract manifold M endowed with an *affine connection* ∇ [5, §5.2] and a *retraction* R, we consider the geometric Newton method in the form proposed by Shub [9, §5] (or see [5, §6.1]). The method consists of iterating the mapping that sends $x \in M$ to $x_+ \in M$ obtained by solving

$$\mathrm{J}_\xi(x)[\eta_x] = -\xi_x, \quad \eta_x \in T_xM, \tag{4.12a}$$

$$x_+ = R_x(\eta_x). \tag{4.12b}$$

Here $\mathrm{J}_\xi(x)$ denotes the Riemannian Jacobian $\mathrm{J}_\xi(x) : T_xM \to T_xM : \zeta_x \mapsto \mathrm{J}_\xi(x)[\zeta_x] = \nabla_{\zeta_x}\xi$.

For the case where M is the quotient $\mathbb{R}_*^{n\times p}/O_p$, we choose the affine connection ∇ defined by

$$\overline{(\nabla_{\eta_{\pi(\mathbf{X})}}\xi)}_{\mathbf{X}} := P^{\mathrm{h}}_{\mathbf{X}}\mathrm{D}\overline{\xi}(\mathbf{X})[\overline{\eta}_{\mathbf{X}}], \tag{4.13}$$

where the overline denotes the horizontal lift. It can be shown that the right-hand side is indeed a horizontal lift and that the definition of ∇ satisfies all the properties of an affine connection. With this choice for ∇, and with a simple choice for the retraction R (see [5, §4.1.2] for the relevant theory), the geometric Newton method on the quotient manifold $\mathbb{R}_*^{n\times p}/O_p$ becomes the iteration that maps $\pi(\mathbf{X})$ to $\pi(\mathbf{X}_+)$ by solving

$$P^{\mathrm{h}}_{\mathbf{X}}\mathrm{D}\overline{\xi}(\mathbf{X})[\overline{\eta}_{\mathbf{X}}] = -\overline{\xi}_{\mathbf{X}}, \quad \overline{\eta}_{\mathbf{X}} \in \mathscr{H}_{\mathbf{X}} \tag{4.14a}$$

$$\mathbf{X}_+ = \mathbf{X} + \overline{\eta}_{\mathbf{X}}, \tag{4.14b}$$

with P^{h} as in (4.11) and $\mathscr{H}_{\mathbf{X}}$ as in (3.7). Note that, in spite of possibly unfamiliar notation, (4.14) only involves basic calculus of functions between matrix spaces.

The local superlinear convergence result for the geometric Newton method (4.14) follows directly from the local convergence result of the general geometric Newton method (4.12), see [5, §6.3]. It states that the iteration converges locally quadratically to the nondegenerate zeros of ξ. Since we have "quotiented out" the symmetry of F by the action of O_p to obtain ξ, it is now reasonable to hope that the zeros of ξ are nondegenerate. As we show in the next section, this is indeed the case for Oja's vector field (4.1b) under the assumption (4.2) of non-repeated eigenvalues.

Note that the proposed geometric Newton method differs from the "canonical" Riemannian Newton algorithm [125] on the quotient $\mathbb{R}_*^{n\times p}/O_p$, because the affine connection chosen is not the Riemannian connection on $\mathbb{R}_*^{n\times p}/O_p$ endowed with the Riemannian metric inherited from the canonical metric in $\mathbb{R}_*^{n\times p}$. However, the property of local superlinear convergence to the nondegenerate zeros still holds (see [9] or [5]).

4.4 A geometric Newton method for Oja's vector field

In this section, we apply the geometric Newton method on $\mathbb{R}_*^{n\times p}/O_p$, given in (4.14), to the case where the tangent vector field ξ on $\mathbb{R}_*^{n\times p}/O_p$ is defined by the horizontal lift

$$\overline{\xi}_{\mathbf{X}} := P^{\mathrm{h}}_{\mathbf{X}}(F(\mathbf{X})), \tag{4.15}$$

with P^{h} as in (4.11) and F as in (4.1b). The resulting Newton iteration is formulated in Algorithm 2. Recall the definitions of $\mathbb{R}_*^{n\times p}$ (3.2), P^{h} (4.11), skew (3.4), $\mathscr{H}_{\mathbf{X}}$ (3.7).

Algorithm 2 Geometric Newton for Oja's vector field

Require: Symmetric positive-definite $n\times n$ matrix $\mathbf{A}$; positive integer $p<n$.
Input: Initial iterate $\mathbf{X}_0\in\mathbb{R}_*^{n\times p}$, i.e., $\mathbf{X}_0$ is a real $(n\times p)$-matrix with full rank.
Output: Sequence of iterates $(\mathbf{X}_k)\subset\mathbb{R}_*^{n\times p}$.

1: **for** $k=0,1,2,\ldots$ **do**
2: Solve the linear system of equations (we drop the subscript k for convenience)

$$\begin{aligned}&P^{\mathrm{h}}_{\mathbf{X}}(\mathbf{AZ}-\mathbf{ZX}^T\mathbf{AX}-\mathbf{XZ}^T\mathbf{AX}-\mathbf{XX}^T\mathbf{AZ}-\mathbf{Z}\mathrm{skew}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{AX})\\&-\mathbf{X}\mathrm{skew}(-(\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{Z}+\mathbf{Z}^T\mathbf{X})(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{AX}+(\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{Z}^T\mathbf{AX}+\mathbf{X}^T\mathbf{AZ})))\\&\qquad=-\left(\mathbf{AX}-\mathbf{XX}^T\mathbf{AX}-\mathbf{X}\mathrm{skew}((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{AX})\right)\end{aligned} \tag{4.16}$$

for the unknown $\mathbf{Z}\in\mathscr{H}_{\mathbf{X}}$.
3: Set

$$\mathbf{X}_{k+1}=\mathbf{X}_k+\mathbf{Z}.$$

4: **end for**

Observe that Algorithm 2 is stated as an iteration in the total space $\mathbb{R}_*^{n\times p}$ of the quotient $\mathbb{R}_*^{n\times p}/O_p$. Formally, the sequence of iterates of the Newton method on $\mathbb{R}_*^{n\times p}/O_p$, for an initial point $\pi(\mathbf{X}_0)\in\mathbb{R}_*^{n\times p}/O_p$, is given by $(\pi(\mathbf{X}_k))_{k\in\mathbb{N}}$, where π is the quotient map (3.6) and $(\mathbf{X}_k)_{k\in\mathbb{N}}$ is the sequence of iterates generated by Algorithm 2.

We point out that (4.16) is merely a linear system of equations. It can be solved by means of iterative solvers that can handle linear systems in operator form. Moreover, these solvers can be stopped early to avoid unnecessary computational effort when the iterate $\mathbf{X}_k$ is still far away from the solution. Guidelines for stopping the linear system solver can be found, e.g., in [49]. In our numerical experiments (Section 4.5) we have used MATLAB's GMRES solver.

For small-scale problems, it is more reasonable to solve the system using direct methods, e.g., Gaussian elimination. Iterative algorithms are to be preferred for large-scale problems and especially if the matrix of the system is sparse or structured, e.g., having Hankel or Toeplitz structure.

Algorithm 2 converges locally quadratically to the nondegenerate zeros of ξ. We first characterize the zeros of ξ, then we show that they are all nondegenerate under the assumption (4.2).

First note that $\xi_{\pi(\mathbf{X})} = 0$ if and only if $P^{\mathrm{h}}_{\mathbf{X}}(F(\mathbf{X})) = \mathbf{0}$, where $\mathbf{X} \in \mathbb{R}^{n\times p}_*$. Under the assumption that $\mathbf{X} \in \mathbb{R}^{n\times p}_*$, the following statements are equivalent:

$$P^{\mathrm{h}}_{\mathbf{X}}(F(\mathbf{X})) = \mathbf{0},$$

$$F(\mathbf{X}) \in \mathrm{im}(P^{\mathrm{a}}_{\mathbf{X}}),$$

$$F(\mathbf{X}) = \mathbf{X}\Omega \text{ for some } \Omega = -\Omega^T,$$

$$\mathbf{A}\mathbf{X} - \mathbf{X}\mathbf{X}^T\mathbf{A}\mathbf{X} = \mathbf{X}\Omega \text{ for some } \Omega = -\Omega^T,$$

$$\mathbf{A}\,\mathrm{col}(\mathbf{X}) \subseteq \mathrm{col}(\mathbf{X}) \text{ and } (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{A}\mathbf{X} - \mathbf{X}^T\mathbf{A}\mathbf{X} \text{ is skew-symmetric},$$

$$\mathbf{A}\,\mathrm{col}(\mathbf{X}) \subseteq \mathrm{col}(\mathbf{X}) \text{ and } (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{A}\mathbf{X} + \mathbf{X}^T\mathbf{A}\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} = 2\mathbf{X}^T\mathbf{A}\mathbf{X}.$$

We thus have an equation of the form $\mathbf{Y}\mathbf{B} + \mathbf{B}\mathbf{Y} = 2\mathbf{B}$, where $\mathbf{B} := \mathbf{X}^T\mathbf{A}\mathbf{X}$ is symmetric positive definite, hence its eigenvalues β_i, $i = 1,\ldots,p$, are all real and positive. The Sylvester operator $\mathbf{Y} \mapsto \mathbf{Y}\mathbf{B} + \mathbf{B}\mathbf{Y}$ is a linear operator whose eigenvalues are $\beta_i + \beta_j$, $i = 1,\ldots,p$, $j = 1,\ldots,p$ [55, § VI]. All these eigenvalues are real and positive, thus nonzero, hence the operator is nonsingular, from which it follows that the equation $\mathbf{Y}\mathbf{B} + \mathbf{B}\mathbf{Y} = 2\mathbf{B}$ has one and only one solution $\mathbf{Y}$. It is readily checked that this unique solution is $\mathbf{Y} = \mathbf{I}$. We have thus shown that $\mathbf{X}^T\mathbf{X} = \mathbf{I}$. The result is summarized in the following proposition.

Proposition 4.4.1 *Let P^{h} be as in* (4.11) *and let F be Oja's vector field* (4.1b)*. Then $\mathbf{X} \in \mathbb{R}^{n\times p}_*$ is a zero of the projected Oja vector field $P^{\mathrm{h}}(F)$ if and only if $\mathbf{A}\,\mathrm{col}(\mathbf{X}) \subseteq \mathrm{col}(\mathbf{X})$ and $\mathbf{X}^T\mathbf{X} = \mathbf{I}$.*

In other words, the full-rank zeros of the projected Oja vector field $P^{\mathrm{h}}(F)$ are the full-rank zeros of Oja's vector field F. This means that we do not lose information by choosing to search for the zeros of ξ (4.15) instead of F (4.1b).

It remains to show that the zeros of ξ are nondegenerate. Let $\pi(\mathbf{X}_*)$ be a zero of ξ, which means that $\mathbf{A}\mathbf{X}_* = \mathbf{X}_*\mathbf{X}_*^T\mathbf{A}\mathbf{X}_*$ and $\mathbf{X}_*^T\mathbf{X}_* = \mathbf{I}$. The task is to show that the Jacobian operator $\mathrm{J}_\xi(\pi(\mathbf{X}_*))$, or equivalently its lifted counterpart

$$\overline{\mathrm{J}_\xi}(\mathbf{X}_*) : \mathrm{im}(P^{\mathrm{h}}) \to \mathrm{im}(P^{\mathrm{h}}) : \mathbf{Z} \mapsto P^{\mathrm{h}}_{\mathbf{X}_*}\left(\mathrm{D}(P^{\mathrm{h}}(F))(\mathbf{X}_*)[\mathbf{Z}]\right), \tag{4.17}$$

is nonsingular. To this end, consider the operator

$$\mathscr{J} : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p} : \mathbf{Z} \mapsto \mathrm{D}(P^{\mathrm{h}}(F))(\mathbf{X}_*)[\mathbf{Z}]. \tag{4.18}$$

Note that $\overline{\mathrm{J}_\xi}(\mathbf{X}_*)$ is the restriction of $\mathscr{J}$ to $\mathrm{im}(P^{\mathrm{h}})$. Consider the decomposition $\mathbb{R}^{n\times p} = \mathrm{im}(P^{\mathrm{p}}) \oplus \mathrm{im}(P^{\mathrm{s}}) \oplus \mathrm{im}(P^{\mathrm{a}})$ and recall that $\mathrm{im}(P^{\mathrm{h}}) = \mathrm{im}(P^{\mathrm{p}}) \oplus \mathrm{im}(P^{\mathrm{s}})$. We show that the corresponding block decomposition of $\mathscr{J}$ is as follows:

$$\begin{bmatrix} * & 0 & 0 \\ ? & * & 0 \\ ? & ? & 0 \end{bmatrix},$$

where "*" denotes nonsingular operators. It then directly follows that the upper two-by-two block of $\mathscr{J}$, which corresponds to $\overline{\mathrm{J}_\xi}(\mathbf{X}_*)$, is nonsingular.

We show that the 11 block (i.e., the "pp" block) is nonsingular. This block is the operator from $\mathrm{im}(P^{\mathrm{p}}_{\mathbf{X}_*})$ to $\mathrm{im}(P^{\mathrm{p}}_{\mathbf{X}_*})$ given by

$$\mathbf{Z} \;\mapsto\; P^{\mathrm{p}}_{\mathbf{X}_*}\,\mathrm{D}(P^{\mathrm{h}}(F))(\mathbf{X}_*)[\mathbf{Z}] = P^{\mathrm{p}}_{\mathbf{X}_*}\mathbf{A}\mathbf{Z} - P^{\mathrm{p}}_{\mathbf{X}_*}\mathbf{Z}\mathbf{X}_*^T\mathbf{A}\mathbf{X}_*.$$

In obtaining this result, we have used the relations $\mathbf{X}_*^T\mathbf{X}_* = \mathbf{I}$, $\mathrm{skew}(\mathbf{X}_*^T\mathbf{A}\mathbf{X}_*) = \mathbf{0}$, $P^{\mathrm{p}}_{\mathbf{X}_*}\mathbf{X}_* = \mathbf{0}$. In view of the hypothesis that the eigenvalues of $\mathbf{A}$ are all simple, this operator is known to be nonsingular; see, e.g., [89, 7].

We show that the 22 block (i.e., the "ss" block) is nonsingular. This block is the operator from $\mathrm{im}(P^{\mathrm{s}}_{\mathbf{X}_*})$ to $\mathrm{im}(P^{\mathrm{s}}_{\mathbf{X}_*})$

$$\mathbf{X}_*\mathbf{S} \quad\mapsto\quad P^{\mathrm{s}}_{\mathbf{X}_*}\Big(\mathrm{D}(P^{\mathrm{h}}(F))(\mathbf{X}_*)[\mathbf{X}_*\mathbf{S}]\Big) = -\mathbf{X}_*(\mathbf{S}\mathbf{X}_*^T\mathbf{A}\mathbf{X}_* + \mathbf{X}_*^T\mathbf{A}\mathbf{X}_*\mathbf{S}).$$

We have used the relation $\mathbf{A}\mathbf{X}_* = \mathbf{X}_*\mathbf{X}_*^T\mathbf{A}\mathbf{X}_*$ to obtain this expression. In view of the previous discussion on the Sylvester operator, this operator is nonsingular.

This completes the proof that the zeros of ξ are nondegenerate. Consequently, for all $\mathbf{X}_0$ sufficiently close to some $\mathscr{S}_i$ (4.6), the sequence $(\mathbf{X}_k)$ generated by Algorithm 2 is such that $\mathbf{X}_k O_p$ converges quadratically to $\mathscr{S}_i$. Recall that $\mathscr{S}_i$ is the set of all orthonormal matrices whose column space is the ith invariant subspace of $\mathbf{A}$.

4.5 Numerical experiments

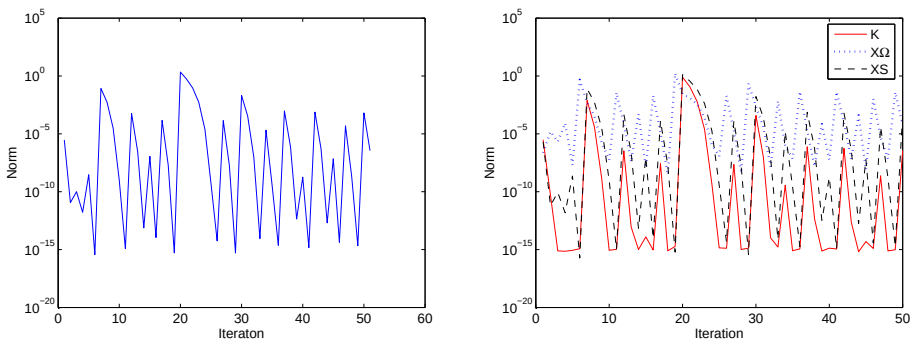
In this section, we report on numerical experiments for both the plain Newton and the geometric Newton method, derived in Section 4.2 and Section 4.4, respectively. The experiments were run using MATLAB. The machine precision is approximately $2\cdot 10^{-16}$.

As mentioned in Section 4.1, the plain Newton method performs poorly due to the fact that the zeros of the cost function F are not isolated. To illustrate this, we consider a

symmetric positive definite matrix $\mathbf{A} \in \mathbb{R}^{6\times 6}$ with uniformly distributed eigenvalues over the interval $[0,1]$ and 100 different initial iterates $\mathbf{X}_0 \in \mathbb{R}^{6\times 3}$. Each of the initial iterates is computed as

$$\mathbf{X}_0 = \mathbf{X}_* + 10^{-6}\mathbf{E}\,,$$

where $\mathbf{X}_*$ is such that $F(\mathbf{X}_*) = \mathbf{0}$ and $\mathbf{E}$ is a (6×3)-matrix with random entries, chosen from a normal distribution with zero mean and unit variance. The simulation is stopped after 50 iterations. One representative example is given in Figure 4.1. In Figure 4.1 (a),



(a) F (b) $\mathbf{K}$, $\mathbf{XS}$, $\mathbf{X\Omega}$

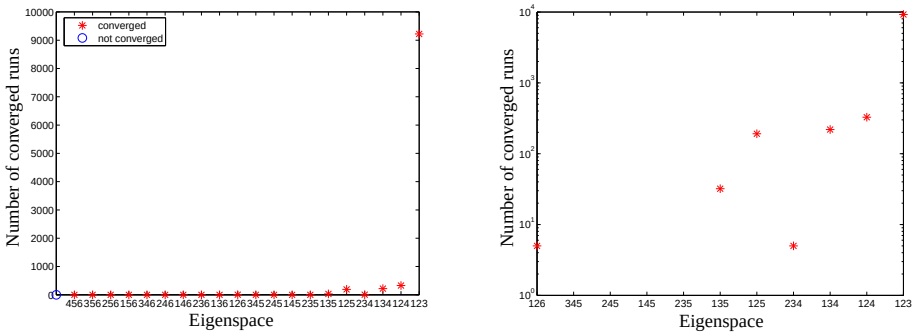
Figure 4.1: *Plain Newton method.*

the norm of $F(\mathbf{X})$ is given for each iteration step. Close to the solution, the system matrix gets singular and the algorithm deviates from the optimal point. In Figure 4.1 (b), we present the evolution of the norms of the three components $\mathbf{K}, \mathbf{X\Omega}$, and $\mathbf{XS}$ of the update vector $\mathbf{Z}$,

$$\mathbf{Z} = \mathbf{X}_\perp \mathbf{K} + \mathbf{X\Omega} + \mathbf{XS}\,,$$

where $\mathbf{X}_\perp$ is the orthogonal complement of $\mathbf{X}$, $\mathbf{\Omega}$ is a skew-symmetric matrix and $\mathbf{S}$ is a symmetric matrix. We see that, even when the $\mathbf{K}$ and $\mathbf{XS}$ component are very small, $\mathbf{X\Omega}$ is quite large. This concords with the fact that the kernel of the Hessian at a stationary point $\mathbf{X}$ is $\{\mathbf{X\Omega} : \mathbf{\Omega}^T = -\mathbf{\Omega}\}$.

Next, we study the geometric Newton method, derived in Section 4.4. Again, we consider $n = 6$, $p = 3$ and a symmetric positive definite matrix $\mathbf{A} \in \mathbb{R}^{6\times 6}$ with uniformly distributed eigenvalues over the interval $[0,1]$. We perform 10^4 experiments with a single matrix $\mathbf{A}$ but with different initial iterates $\mathbf{X}_0 \in \mathbb{R}^{6\times 3}_*$ with random entries, chosen from a normal distribution with zero mean and unit variance. In Figure 4.2, we show the number of runs that converged to each of the eigenspaces of $\mathbf{A}$. The dominant eigenspace is marked by "123". In general, "ijk" stands for the eigenspace spanned by the i^{th}, j^{th}, and k^{th}



(a) Linear plot (b) Logarithmic scale

Figure 4.2: *Geometric Newton method.*

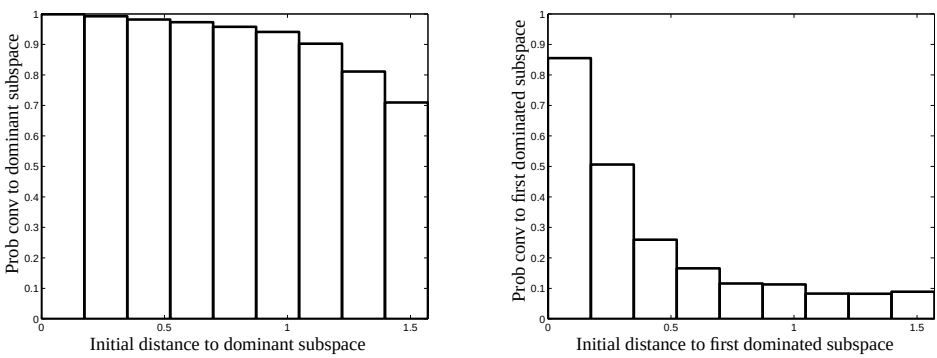
eigenvectors[1]. Let $\mathbf{W}$ be the matrix of all eigenvectors. We declare that the algorithm has converged to eigenspace "ijk" when the largest principal angle [57, §12.4.3] between this eigenspace and $\mathbf{X}$ is smaller than 10^{-10} after 50 iterations. In our experiment, all the runs have converged to one of the 20 possible eigenspaces. In general, there may be cases where the algorithm does not converge to any of the eigenspaces. This may occur when the initial iterate $\mathbf{X}_0$ is very close to the boundary of one of the basins of attraction. However, these cases are rare.

It appears that the basin of attraction of the dominant eigenspace is the largest. This finding was not expected *a priori*, but it is not in contradiction with any known property of Algorithm 2. Note in particular that, in contrast with the Grassmann-based Newton method for the generalized Rayleigh quotient investigated in [48, 89, 4], Algorithm 2 is not invariant by reflections of the spectrum of $\mathbf{A}$; namely, Algorithm 2 is not invariant by transformations of the type $\mathbf{A} \mapsto -(\mathbf{A} - \mu\mathbf{I})$, $\mu \in \mathbb{R}$. Therefore, whereas the Grassmann-based method does not favor one end of the spectrum over the other, there is no reason why Algorithm 2 should display a similar behavior. Depending on the application, the observed global behavior of Algorithm 2, with a large basin of attraction for the dominant eigenspace, can be considered an advantage over the Grassmann-based iteration.

To investigate this issue further, we performed numerical experiments to assess the distribution of the area of the various basins of attraction, measured in terms of the largest principal angle between $\mathrm{col}(\mathbf{X}(0))$ and eigenspaces of interest. It is known (see, e.g., [2]) that if the elements of $\mathbf{X}(0)$ are drawn from a Gaussian distribution, then the bins corresponding to small values of the largest principal angle are scarcely populated, hence the statistics for these bins are unreliable. To remedy this, it is possible to draw $\mathbf{X}(0)$ from a Gaussian distribution and discard some of them according to the probability density

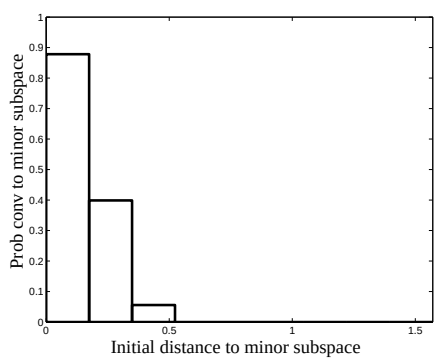
[1]For convenience, we consider an eigenvalue decomposition, where the eigenvalues of $\mathbf{A}$ are put in descending order.

function given in [2], so as to obtain a uniform distribution on a prescribed subinterval of $[0, \pi/2]$ for the value of the largest principal angle. However, to obtain a uniform distribution on a reasonably large interval, we had to discard so many $\mathbf{X}(0)$'s that this approach turned out to be impractical. Therefore, in the numerical experiments mentioned below, we drew $\mathbf{X}(0)$ according to $\mathbf{X}(0) = \mathbf{X}_* + a\mathbf{E}$, where a is uniformly distributed over the interval $[0,1]$ and the elements of $\mathbf{E}$ are drawn from the Gaussian distribution with mean zero and standard deviation one; this technique resulted in bins that are reasonably evenly populated. We declare that the iteration has converged to eigenspace $\mathscr{O}_i$ if the largest principal angle between $\mathrm{col}(\mathbf{X}(k))$ and $\mathscr{O}_i$ is smaller than 10^{-10} when $k = 50$. The distribution of the area of the basins of attraction corresponding to the dominant eigenspace, to the first dominated eigenspace (i.e., eigenspace 124), and to the minor eigenspace, are shown on the histograms of Figure 4.3, obtained over 10^4 experiments. We observed that the sequences that do not converge to the dominant eigenspace tend to



(a) Dominant eigenspace

(b) First dominated eigenspace



(c) Minor eigenspace

Figure 4.3: *Distribution of the basin of attraction of three eigenspaces of interest. The horizontal axis shows the largest principal angle between* $\mathrm{col}(\mathbf{X}(0))$ *and the eigenspace under consideration. The vertical axis gives the fraction of initial conditions that led to convergence to the eigenspace.*

display a rather large first step, which suggests that they are initialized close to a point where the Jacobian is not invertible.

In interpreting the results of Figure 4.3, one should bear in mind that Algorithm 2 induces an iteration on $\mathbb{R}_*^{n\times p}/O_p$, but not on the Grassmann manifold. Hence, the largest principal angle to $\mathrm{col}(\mathbf{X}(0))$ is not a measure of distance to the solution; for example, if the column space of $\mathbf{X}(0)$ is close to the dominant eigenspace but $\mathbf{X}(0)$ is far from orthonormality, then $\mathbf{X}(0)$ is far away from any zero of Oja's vector field even though $\mathbf{X}(0)$ may be tallied in the leftmost bin of the first histogram.

Finally, the superlinear convergence rate of the algorithm is illustrated in Figure 4.4.

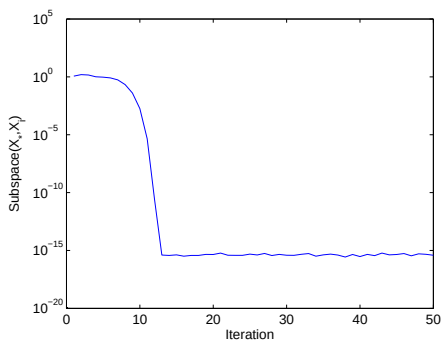


Figure 4.4: *Illustration of the superlinear convergence of the geometric Newton method.*

4.6 Conclusions

We have investigated the use of Newton's method to compute superlinearly the zeros of Oja's vector field (4.1b). Due to a symmetry in the vector field by the action of the orthogonal group, its zeros are never isolated, which causes the plain Newton method to behave poorly. We have proposed a remedy that consists in "quotienting out" the symmetry. This led to the formulation of a geometric Newton algorithm that seeks the zeros of a projection of Oja's vector field. We have shown that the zeros of the projected vector field are the same as the zeros of the original vector field. Moreover, these zeros are nondegenerate. This means that by quotienting out the action of the orthogonal group, we have removed just enough symmetry to make the zeros nondegenerate. In view of the nondegeneracy property, it follows directly from the convergence theory of the abstract geometric Newton method that the resulting algorithm converges locally superlinearly to the zeros of Oja's vector field.

Invariant subspace computation has been and still is a very active area of research. As a method for invariant subspace computation, it is doubtful that the proposed algorithm can outperform the state-of-the-art methods. In particular, the Grassmann-based approach [48, 89, 4], that can be thought of as quotienting out the action of the whole general linear group instead of the orthogonal group, leads to a Newton equation that lives in a smaller subspace of $\mathbb{R}^{n\times p}$ and that can be solved in fewer flops. When $n \gg p$, however, the number of operations necessary to apply the linear operator in the Newton equation is of the same order. Moreover, the Grassmann-based algorithm does not possess the remarkable feature of naturally converging towards orthonormal matrices, i.e., without having to resort to orthogonalization steps such as Gram-Schmidt.

The quotient manifold $\mathbb{R}_*^{n\times p}/O_p$ seems to have received little attention in the literature, in contrast to the more famous Grassmann and Stiefel manifolds. In the next chapter, we will show how it can be used in the context of the low multilinear rank approximation of higher-order tensors.

Chapter 5

Geometric Newton algorithm

In this chapter, we propose a Newton algorithm for the best low multilinear rank approximation of higher-order tensors. The solutions of the optimization problem (2.3) are reformulated as zeros of a suitable function. However, the symmetry property (3.1) by the action of the orthogonal group remains, the zeros of the new function are not isolated, which means that the plain Newton method is expected to have difficulties. Following Chapter 4 and [3], we propose a Newton method that makes use of differential-geometric techniques for quotient manifolds.

An advantage of the algorithm is its fast convergence rate in the neighborhood of the solution. Although global convergence can not be guaranteed, starting from a good initial point can improve the chances of converging to a solution. We illustrate the quadratic convergence and compare the algorithm with HOOI (Algorithm 1, Section 2.4.2). For simplicity, we only consider third-order tensors. The generalization to tensors of order higher than three is feasible.

For information on differential-geometric versions of Newton's method, we refer the reader to [62, 135, 11, 45, 88]. As in Chapter 4, here we rely to a large extent on the theory developed in [9, 5]. Recently, a Newton-Grassmann method has been suggested in [51]. It works on the Grassmann manifold whereas the algorithm proposed in this chapter [72] is a generalization of the concepts proposed in Chapter 4 and [3]. Finally, we mention that quasi-Newton methods on Grassmannians have been developed in [118].

5.1 Reformulation of the problem

We first formulate the solution of the optimization problem (2.3) as a zero of a function F. The function F has a symmetry property by the action of the orthogonal group, which is

known to be a source of difficulty when the usual Newton method is employed to compute the zeros of F [3].

5.1.1 The new cost function

It follows from (2.3) that the matrix $\mathbf{U} \in St(R_1, I_1)$ is optimal if and only if [62, Th. 3.17] its columns span the same subspace as the R_1 dominant left singular vectors of the matrix $\mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W})$. A necessary condition for this is that the column space of $\mathbf{U}$ is an invariant subspace of $\mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W})(\mathbf{V} \otimes \mathbf{W})^T \mathbf{A}_{(1)}^T$, i.e.,

$$\mathbf{U}\,\mathbf{S}_1 = \mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W})(\mathbf{V} \otimes \mathbf{W})^T \mathbf{A}_{(1)}^T \mathbf{U} \tag{5.1}$$

for some $\mathbf{S}_1 \in \mathbb{R}^{R_1 \times R_1}$. Similar equations can be derived for $\mathbf{V}$ and $\mathbf{W}$, which leads to the following set of equations:

$$\begin{aligned} \mathbf{U}\,\mathbf{S}_1 &= \mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W})(\mathbf{V} \otimes \mathbf{W})^T \mathbf{A}_{(1)}^T \mathbf{U}, \\ \mathbf{V}\,\mathbf{S}_2 &= \mathbf{A}_{(2)}(\mathbf{W} \otimes \mathbf{U})(\mathbf{W} \otimes \mathbf{U})^T \mathbf{A}_{(2)}^T \mathbf{V}, \\ \mathbf{W}\,\mathbf{S}_3 &= \mathbf{A}_{(3)}(\mathbf{U} \otimes \mathbf{V})(\mathbf{U} \otimes \mathbf{V})^T \mathbf{A}_{(3)}^T \mathbf{W}, \end{aligned} \tag{5.2}$$

for some $\mathbf{S}_1 \in \mathbb{R}^{R_1 \times R_1}$, $\mathbf{S}_2 \in \mathbb{R}^{R_2 \times R_2}$ and $\mathbf{S}_3 \in \mathbb{R}^{R_3 \times R_3}$. We define

$$\begin{aligned} \mathbf{R}_1(\mathbf{X}) &= \mathbf{U}^T \mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W}), \\ \mathbf{R}_2(\mathbf{X}) &= \mathbf{V}^T \mathbf{A}_{(2)}(\mathbf{W} \otimes \mathbf{U}), \\ \mathbf{R}_3(\mathbf{X}) &= \mathbf{W}^T \mathbf{A}_{(3)}(\mathbf{U} \otimes \mathbf{V}), \end{aligned}$$

where $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$. We further define a Euclidean space $\mathscr{E}$ by

$$\mathscr{E} = \mathbb{R}^{I_1 \times R_1} \times \mathbb{R}^{I_2 \times R_2} \times \mathbb{R}^{I_3 \times R_3}, \tag{5.3}$$

and the function F by

$$\begin{aligned} &F : \mathscr{E} \to \mathscr{E}, \\ &\quad \mathbf{X} \mapsto (F_1(\mathbf{X}), F_2(\mathbf{X}), F_3(\mathbf{X})), \end{aligned} \tag{5.4}$$

where

$$\begin{aligned} F_1(\mathbf{X}) &= \mathbf{U}\,\mathbf{R}_1(\mathbf{X})\mathbf{R}_1(\mathbf{X})^T - \mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W})\mathbf{R}_1(\mathbf{X})^T, \\ F_2(\mathbf{X}) &= \mathbf{V}\,\mathbf{R}_2(\mathbf{X})\mathbf{R}_2(\mathbf{X})^T - \mathbf{A}_{(2)}(\mathbf{W} \otimes \mathbf{U})\mathbf{R}_2(\mathbf{X})^T, \\ F_3(\mathbf{X}) &= \mathbf{W}\mathbf{R}_3(\mathbf{X})\mathbf{R}_3(\mathbf{X})^T - \mathbf{A}_{(3)}(\mathbf{U} \otimes \mathbf{V})\,\mathbf{R}_3(\mathbf{X})^T. \end{aligned}$$

We will look for $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$ such that

$$F(\mathbf{X}) = \mathbf{0},$$

from which (5.2) follows.

5.1.2 The invariance property

Notice that the functions F_i, $i = 1,2,3$ have the invariance property

$$F_i(\mathbf{XQ}) = F_i(\mathbf{X})\,\mathbf{Q}_i, \tag{5.5}$$

where $\mathbf{XQ} = (\mathbf{UQ}_1, \mathbf{VQ}_2, \mathbf{WQ}_3)$ and $\mathbf{Q}_i \in O_{R_i}, i = 1,2,3$ are orthogonal matrices. Thus,

$$F(\mathbf{X}) = \mathbf{0} \quad \Longleftrightarrow \quad F(\mathbf{XQ}) = \mathbf{0}.$$

This means that the zeros of F are not isolated. Because of this, a plain Newton method will most probably perform poorly (see, for example, [5, Prop. 2.1.2], [3] or Chapter 4). We remedy this problem in the next section by a differential-geometric approach.

5.2 Differential-geometric Newton method for the best rank-(R_1, R_2, R_3) approximation

In the spirit of [3] and Chapter 4, we propose a remedy to the symmetry problem (5.5) of F, in the form of a differential-geometric Newton method defined on a quotient space where the symmetry is removed. The general theory on Newton's method on manifolds can be found in [5], in a form that is chiefly based on [9].

Recall Section 3.5 for the main concepts concerning the manifold $\mathbb{R}_*^{n\times p}/O_p$. As in [3] and Chapter 4, we consider the following Newton equation

$$\left|\begin{array}{l} P_{\mathbf{X}}^h \,\mathrm{D}(P_{\mathbf{X}}^h F)(\mathbf{X})[\Delta] = -P_{\mathbf{X}}^h F(\mathbf{X}), \quad \Delta \in \mathscr{H}_{\mathbf{X}} \\ \mathbf{X}_+ = \mathbf{X} + \Delta, \end{array}\right.$$

where $\mathrm{D}(P_{\mathbf{X}}^h F)(\mathbf{X})[\Delta]$ is the derivative of $P_{\mathbf{X}}^h F(\mathbf{X})$ along Δ and $\mathbf{X}_+$ stands for the next iterate. However, in the case of function (5.4), we need to work on the product manifold

$$M = \mathbb{R}_*^{I_1\times R_1}/O_{R_1} \times \mathbb{R}_*^{I_2\times R_2}/O_{R_2} \times \mathbb{R}_*^{I_3\times R_3}/O_{R_3} \tag{5.6}$$

instead of the simpler manifold $\mathbb{R}_*^{n\times p}/O_p$. Hence, we need to develop a nontrivial generalization of the algorithm proposed in [3] and Chapter 4, which involves more difficult computations. The inner product of two elements on the horizontal space

$$\mathscr{H}_{(\mathbf{U},\mathbf{V},\mathbf{W})} = \mathscr{H}_{\mathbf{U}} \times \mathscr{H}_{\mathbf{V}} \times \mathscr{H}_{\mathbf{W}},$$

where each component of the right-hand side is as in (3.7), is defined as the sum of the three inner products (3.9) of the corresponding components. The orthogonal projection is now

$$P_{(\mathbf{U},\mathbf{V},\mathbf{W})}(\mathbf{Z}_{\mathbf{U}}, \mathbf{Z}_{\mathbf{V}}, \mathbf{Z}_{\mathbf{W}}) = (P_{\mathbf{U}}^h(\mathbf{Z}_{\mathbf{U}}), P_{\mathbf{V}}^h(\mathbf{Z}_{\mathbf{V}}), P_{\mathbf{W}}^h(\mathbf{Z}_{\mathbf{W}})), \tag{5.7}$$

where P^h is as in (3.8). P projects the columns of $\mathbf{Z_U}, \mathbf{Z_V}$ and $\mathbf{Z_W}$ onto the orthogonal complements of the column spaces of $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$, respectively. Finally, for the retraction R of a given tangent vector $(\mathbf{Z_U}, \mathbf{Z_V}, \mathbf{Z_W})$ at a point $(\mathbf{U}, \mathbf{V}, \mathbf{W})$, based on (3.11), we use the following definition:

$$R_{(\mathbf{U},\mathbf{V},\mathbf{W})}(\mathbf{Z_U}, \mathbf{Z_V}, \mathbf{Z_W}) = ((\mathbf{U}+\mathbf{Z_U}), (\mathbf{V}+\mathbf{Z_V}), (\mathbf{W}+\mathbf{Z_W}))\,. \tag{5.8}$$

Then, we have the following Newton equation

$$\left|\begin{array}{l} P^h_{\mathbf{U}}\,\mathrm{D}(P^h_{\mathbf{U}}F_1)(\mathbf{X})[\Delta] = -P^h_{\mathbf{U}}F_1(\mathbf{X}) \\ P^h_{\mathbf{V}}\,\mathrm{D}(P^h_{\mathbf{V}}F_2)(\mathbf{X})[\Delta] = -P^h_{\mathbf{V}}F_2(\mathbf{X})\,, \quad \Delta = (\Delta_1, \Delta_2, \Delta_3) \in \mathscr{H}_{\mathbf{U}} \times \mathscr{H}_{\mathbf{V}} \times \mathscr{H}_{\mathbf{W}}\,. \\ P^h_{\mathbf{W}}\,\mathrm{D}(P^h_{\mathbf{W}}F_3)(\mathbf{X})[\Delta] = -P^h_{\mathbf{W}}F_3(\mathbf{X}) \\ \mathbf{X}_+ = \mathbf{X}+\Delta \end{array}\right. \tag{5.9}$$

In order to compute $\mathrm{D}(P^h_{\mathbf{U}}F_1)(\mathbf{X})[\Delta]$, we first compute

$$\begin{aligned} P^h_{\mathbf{U}}F_1(\mathbf{X}) \;=\; & \mathbf{U}\mathbf{R}_1(\mathbf{X})\mathbf{R}_1(\mathbf{X})^T - \mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T\mathbf{A}_{(1)}^T\mathbf{U} \\ & +\mathbf{U}\mathrm{skew}((\mathbf{U}^T\mathbf{U})^{-1}\mathbf{R}_1(\mathbf{X})\mathbf{R}_1(\mathbf{X})^T)\,. \end{aligned}$$

Here we have used that $\mathrm{skew}(\mathbf{S}) = \mathbf{0}$, when $\mathbf{S}$ is a symmetric matrix. Hence,

$$\begin{aligned} &\mathrm{D}(P^h_{\mathbf{U}}F_1)(\mathbf{X})[\Delta] \\ &\quad= \Delta_1\mathbf{R}_1(\mathbf{X})\mathbf{R}_1(\mathbf{X})^T + \mathbf{U}\Delta_1^T\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})\mathbf{R}_1(\mathbf{X})^T + \mathbf{U}\mathbf{R}_1(\mathbf{X})(\mathbf{V}\otimes\mathbf{W})^T\mathbf{A}_{(1)}^T\Delta_1 \\ &\qquad +\mathbf{U}\mathbf{U}^T\mathbf{A}_{(1)}\mathrm{D}((\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T)(\mathbf{X})[\Delta]\mathbf{A}_{(1)}^T\mathbf{U} \\ &\qquad -\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T\mathbf{A}_{(1)}^T\Delta_1 \\ &\qquad -\mathbf{A}_{(1)}\mathrm{D}((\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T)(\mathbf{X})[\Delta]\mathbf{A}_{(1)}^T\mathbf{U} \\ &\qquad +\Delta_1\mathrm{skew}((\mathbf{U}^T\mathbf{U})^{-1}\mathbf{R}_1(\mathbf{X})\mathbf{R}_1(\mathbf{X})^T) \\ &\qquad +\mathbf{U}\mathrm{skew}(-(\mathbf{U}^T\mathbf{U})^{-1}(\Delta_1^T\mathbf{U}+\mathbf{U}^T\Delta_1)(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{R}_1(\mathbf{X})\mathbf{R}_1(\mathbf{X})^T) \\ &\qquad +\mathbf{U}\mathrm{skew}((\mathbf{U}^T\mathbf{U})^{-1}(\Delta_1^T\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})\mathbf{R}_1(\mathbf{X})^T + \mathbf{R}_1(\mathbf{X})(\mathbf{V}\otimes\mathbf{W})^T\mathbf{A}_{(1)}^T\Delta_1)) \\ &\qquad +\mathbf{U}\mathrm{skew}((\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\mathbf{A}_{(1)}\mathrm{D}((\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T)(\mathbf{X})[\Delta]\mathbf{A}_{(1)}^T\mathbf{U})\,, \end{aligned} \tag{5.10}$$

where

$$\begin{aligned} \mathrm{D}((\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T)(\mathbf{X})[\Delta] \;=\; & (\Delta_2\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T + (\mathbf{V}\otimes\Delta_3)(\mathbf{V}\otimes\mathbf{W})^T \\ & +(\mathbf{V}\otimes\mathbf{W})(\Delta_2\otimes\mathbf{W})^T + (\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\Delta_3)^T\,. \end{aligned}$$

Similar formulas can be obtained for the other two equations in (5.9). Notice that (5.9) is a linear system of equations in the unknown Δ. One possible way to solve it, is by using

MATLAB's GMRES solver. Our numerical experiments in Section 5.3 have been solved in this way. However, as we mentioned in the previous chapter, direct methods are more suitable for small-scale systems. Iterative methods are useful for large-scale systems and especially large-scale systems involving structured matrices.

Finally, we summarize the algorithm in Algorithm 3.

Algorithm 3 Geometric Newton algorithm

1: Take initial estimates $\mathbf{U}_0 \in St(R_1,I_1)$, $\mathbf{V}_0 \in St(R_2,I_2)$, $\mathbf{W}_0 \in St(R_3,I_3)$, $\mathbf{X}_0 = (\mathbf{U}_0,\mathbf{V}_0,\mathbf{W}_0)$. A good choice of $\mathbf{X}_0$ is often obtained from the truncated HOSVD.

2: **for** $k = 0,1,2,\ldots$ **do**

3: Solve the linear system of equations:

$$\left|\begin{array}{l} P^h_{\mathbf{U}}\,\mathrm{D}(P^h_{\mathbf{U}}F_1)(\mathbf{X}_k)[\Delta_k] = -P^h_{\mathbf{U}}F_1(\mathbf{X}_k) \\ P^h_{\mathbf{V}}\,\mathrm{D}(P^h_{\mathbf{V}}F_2)(\mathbf{X}_k)[\Delta_k] = -P^h_{\mathbf{V}}F_2(\mathbf{X}_k) \\ P^h_{\mathbf{W}}\,\mathrm{D}(P^h_{\mathbf{W}}F_3)(\mathbf{X}_k)[\Delta_k] = -P^h_{\mathbf{W}}F_3(\mathbf{X}_k) \end{array}\right. , \quad \Delta_k = (\Delta_{k1},\Delta_{k2},\Delta_{k3}) \in \mathscr{H}_{\mathbf{U}} \times \mathscr{H}_{\mathbf{V}} \times \mathscr{H}_{\mathbf{W}}\,.$$

4: Set

$$\mathbf{X}_{k+1} = \mathbf{X}_k + \Delta_k.$$

5: **end for**

6: $\mathbf{X} = \mathbf{X}_{k+1}$.

7: Compute

$$\mathscr{B} = \mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T,$$

where $\mathbf{X} = (\mathbf{U},\mathbf{V},\mathbf{W})$. The best low multilinear rank approximation $\hat{\mathscr{A}}$ is given by

$$\hat{\mathscr{A}} = \mathscr{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}.$$

The Newton method has local quadratic convergence to the nondegenerate zeros of the vector field ξ on M (5.6) represented by the horizontal lift P^hF. First, observe that if $\mathbf{X}_*$ is a zero of F (5.4), then the equivalence class of $\mathbf{X}_*$ is a zero of ξ. It remains to see what the nondegeneracy condition entails. One would expect that nondegeneracy of the zeros holds under mild conditions, much as in [3] and Chapter 4. However, in contrast to [3] and Chapter 4, a mathematical proof of such a property remains elusive. Instead, we have obtained numerical evidence that nondegeneracy holds under generic conditions. Indeed, over our 10^3 numerical experiments, we have observed that the condition number of the

linear operator

$$\begin{array}{rccl} G: & \mathscr{H}_{(\mathbf{U},\mathbf{V},\mathbf{W})} & \to & \mathscr{H}_{(\mathbf{U},\mathbf{V},\mathbf{W})}, \\ & \Delta & \mapsto & P^h_{\mathbf{X}_*} \mathrm{D}(P^h F(\mathbf{X}_*)[\Delta]), \end{array}$$

where $\mathbf{X}_*$ is a zero of $P^h F$, was always smaller than 10^{10}. The condition number of G was computed as the condition number of the matrix that acts on vectors ($\mathrm{vec}(\Delta)$) to produce $\mathrm{vec}(G(\Delta))$. The columns of that matrix can be obtained by applying the operator on the canonical basis. The fact that the condition number of the matrix was not too small suggests that the zeros of $P^h F$ — viewed as a vector field on M (5.6) — are generically nondegenerate. (This contrasts with the zeros of F (5.4) on the Euclidean space $\mathscr{E}$ (5.3); recall that they are always degenerate in view of the invariance property (5.5).)

5.3 Numerical Experiments

In this section, we summarize our practical experience with Algorithm 3. Moreover, we compare the differential-geometric Newton method with the HOOI algorithm (Algorithm 1, Section 2.4.2).

We first considered tensors $\overline{\mathscr{T}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ that were exactly rank-(R_1, R_2, R_3). We assumed that the data were affected by additive noise. The tensor that was approximated was then given by

$$\mathscr{A}(\sigma) = \mathscr{T} + \sigma * \mathscr{E} = \overline{\mathscr{T}}/\|\overline{\mathscr{T}}\| + \sigma * \overline{\mathscr{E}}/\|\overline{\mathscr{E}}\|, \tag{5.11}$$

in which $\overline{\mathscr{E}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ represents noise with elements uniformly distributed over the interval $[0,1]$ and in which σ controls the noise level. If σ is small, then $\mathscr{A}(\sigma)$ has a dominant rank-(R_1, R_2, R_3) part. If σ is large, then $\mathscr{A}(\sigma)$ is unstructured. Normalization does not change rank properties, so the tensor $\mathscr{T}$ has the same mode-n ranks as $\overline{\mathscr{T}}$, namely, (R_1, R_2, R_3). To impose these rank conditions, $\overline{\mathscr{T}}$ was constructed as the product of a tensor $\mathscr{C} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ and three matrices $\mathbf{M}_i \in \mathbb{R}^{I_i \times R_i}$, $i = 1,2,3$, all with elements uniformly distributed over the interval $[0,1]$, in the following way

$$\overline{\mathscr{T}} = \mathscr{C} \bullet_1 \mathbf{M}_1 \bullet_2 \mathbf{M}_2 \bullet_3 \mathbf{M}_3.$$

In Figure 5.1, we illustrate our experience. The dimensions of $\mathscr{T}$ are $I_1 = I_2 = I_3 = 20$, the mode-n ranks are $(R_1, R_2, R_3) = (5,5,5)$ and $\sigma = 0.1$. The initial matrices $\mathbf{U}_0$, $\mathbf{V}_0$ and $\mathbf{W}_0$ were taken from the truncated HOSVD and 5 iterations of HOOI are performed before starting with the differential-geometric Newton method. The reason for these initial iterations is that a Newton method needs a good starting value. The value obtained from the truncated HOSVD is not always in the basin of attraction of a zero of F [51]. HOOI was then run for 20 additional iterations and the geometric Newton algorithm until the value of the cost function F was close to the machine precision. We present the original

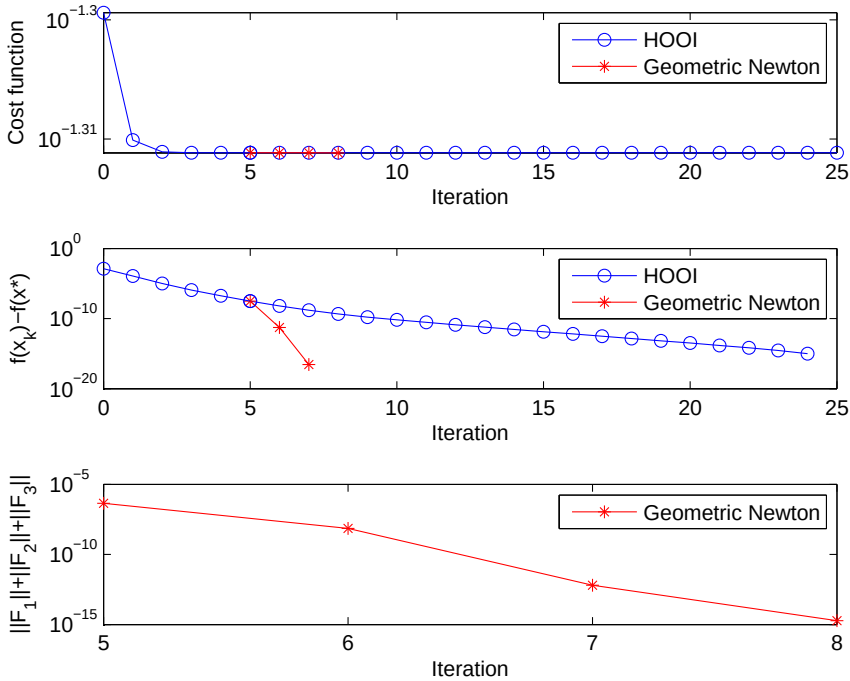


Figure 5.1: *Evolution of the original cost function (2.2), the difference in cost function between each iteration and the last iterate and the function F from (5.4). The tensor $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ is as in (5.11), with $R_1 = R_2 = R_3 = 5$ and $\sigma = 0.1$ The starting values are taken from the truncated HOSVD and initial 5 iterations of HOOI are performed.*

cost function (2.2), the difference in cost function between each iteration and the last iterate and the function F from (5.4). As it can be seen from the figure, the differential-geometric Newton method converges in fewer iterations than HOOI.

Next, we considered tensors $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ with elements uniformly distributed over the interval $[0,1]$ and looked again for the best rank-$(5,5,5)$ approximation of these tensors. We started from values taken from the truncated HOSVD and performed 20 iterations of HOOI. This problem is more difficult than the previous one, so more initial HOOI iterations were necessary before starting Algorithm 3. We ran HOOI for 30 additional iterations and the geometric Newton algorithm until the value of the cost function F was close to the machine precision. In Figure 5.2, the results of this experiment are given for one representative example. The difference between the two algorithms, with respect to the number of iterations necessary for converge, is even greater than in the previous test.

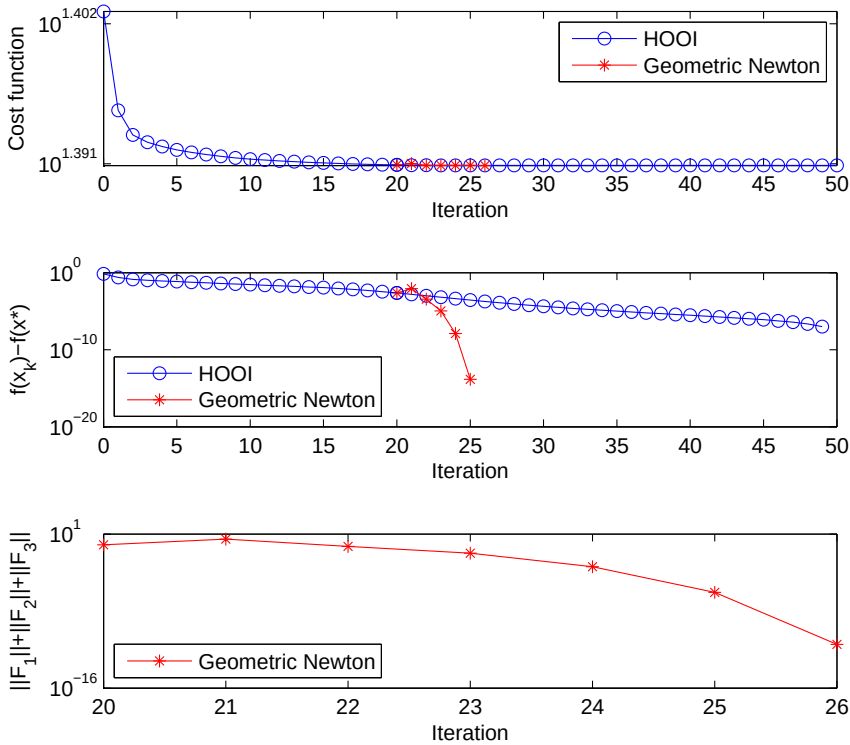


Figure 5.2: *Evolution of the original cost function (2.2), the difference in cost function between each iteration and the last iterate and the function F from (5.4). The tensor* $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ *has elements uniformly distributed over the interval* $[0,1]$ *and* $R_1 = R_2 = R_3 = 5$*. The starting values are taken from the truncated HOSVD and initial 20 iterations of HOOI are performed.*

This seems to be typical for tensors that do not have a clearly dominant rank-(R_1, R_2, R_3) part.

However, in both cases Algorithm 3 needs more time per iteration than HOOI. The main computational cost of one HOOI iteration is $O(I^3R + IR^4 + R^6)$ (see Section 2.4.2), if we assume for simplicity that $R_1 = R_2 = R_3 = R$ and $I_1 = I_2 = I_3 = I$. For the differential-geometric Newton method, the main computational cost comes from solving the linear system of equations (5.9). We use MATLAB's GMRES solver. The products $\mathscr{A} \bullet_1 \mathbf{U}^T$, $\mathscr{A} \bullet_2 \mathbf{V}^T$, $\mathscr{A} \bullet_3 \mathbf{W}^T$, $\mathscr{A} \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T$, etc. need to be computed only once at the beginning of the GMRES step. This can be done with a total cost of $O(I^3R)$ flops. Each step within

GMRES is dominated by the computation of expressions like $\mathbf{A}_{(1)}(\mathbf{V} \otimes \Delta_3)$, see (5.10). Since $\mathscr{A} \bullet_2 \mathbf{V}^T$ is already available, the cost is $O(I^2R^2)$ flops. Note that $\mathscr{A} \bullet_3 \Delta_3^T \bullet_2 \mathbf{V}^T = \mathscr{A} \bullet_2 \mathbf{V}^T \bullet_3 \Delta_3^T$ so $\mathbf{A}_{(1)}(\Delta_2 \otimes \mathbf{W})$ is as expensive as $\mathbf{A}_{(1)}(\mathbf{V} \otimes \Delta_3)$. Here we have used that $\mathbf{V}, \mathbf{W}, \Delta_2$, and Δ_3 have the same dimensions. Assuming that we require an accurate solution of the linear system (5.9), the number of steps within GMRES is comparable to the dimension of the horizontal space (3.7), which is $O(IR)$. Hence, the total cost for one iteration of the differential-geometric Newton method is $O(I^3R) + O(I^2R^2\,IR) = O(I^3R^3)$ flops.

There is a possible disadvantage of the proposed method. As a Newton method, it does not necessarily converge to a local maximum of (2.3). First, not all zeros of F correspond to local maxima of (2.3). Moreover, in theory, Newton's method can diverge. However, we did not observe this behavior in our numerical experiments. To favor convergence to the maximum of (2.3), one can initialize the Newton algorithm by performing an HOSVD followed by a few iterations of HOOI. One could additionally check for the negative definiteness of the Hessian H of (2.3). Indeed, if H is negative definite then the converged point is a local maximum. This was the case in both experiments shown in Figure 5.1 and Figure 5.2.

5.4 Conclusions

In this chapter, we have developed a differential-geometric method for computing the best rank-(R_1, R_2, R_3) approximation of a tensor. It is a Newton method based on quotient manifold techniques. The advantage of the algorithm is its fast convergence in the neighborhood of the solution. On the other hand, *global* convergence to the best approximation of the tensor is not guaranteed. However, starting from a good initial guess and performing several iterations of HOOI increases the chances of converging to a solution.

In the next two chapters we develop algorithms that combine advantages of HOOI and the differential-geometric Newton method, namely, convergence to a local minimum of (2.2) with a convergence rate that is faster than linear. We will consider both trust-region and conjugate gradient based algorithms on a quotient manifold.

Chapter 6

Trust-region algorithm

In this chapter, we develop an algorithm based on the trust-region scheme on Riemannian manifolds. The tensor approximation problem is expressed as a minimization of a cost function on a product of three Grassmann manifolds. We apply the Riemannian trust-region scheme, using the truncated conjugate gradient method for solving the trust-region subproblem. Making use of second order information of the cost function, superlinear convergence is achieved. If the stopping criterion of the subproblem is chosen adequately, the local convergence rate is quadratic. We compare this new method with HOOI and discuss the advantages over Newton-type methods.

The main advantage over HOOI of our new algorithm is its superlinear, in particular quadratic, convergence speed. The convergence speed of HOOI is only linear. On the other hand, the HOOI iterations are cheaper in general. However, if the multilinear rank is small compared to the tensor dimensions, which is often the case in practice, the cost per iteration of both algorithms is comparable. A detailed comparison of these two algorithms is given in Section 6.3. Compared to the Newton-type algorithms, the advantage of the proposed algorithm is its more stable behavior. For the trust-region algorithm, convergence to stationary points is guaranteed for all initial points. This is not the case for the Newton-type methods. Moreover, in practice, the trust-region algorithm converges to a local minimum, whereas the Newton-type methods may converge to any type of stationary point (minimum, maximum or saddle point).

In Section 6.1, the basics of the trust-region method in $\mathbb{R}^n$, in a Euclidean space and on a Riemannian manifold are given. We refer to [1] for the theory of the Riemannian trust-region method and to the GENRTR[1] package for a generic MATLAB implementation. Our trust-region based algorithm for the best rank-(R_1, R_2, R_3) approximation of third-

[1] `http://www.math.fsu.edu/~cbaker/GenRTR/`

order tensors is described in Section 6.2. A summary of the first version of the algorithm has been proposed in [70]. The algorithm is described in detail in [71]. Section 6.3 makes a comparison between the trust-region algorithm and HOOI. A comparison with Newton-type methods is made in Section 6.4. We draw our conclusions in Section 6.5.

6.1 Riemannian trust-region scheme

In this section, we recall the idea behind trust-region methods. We consider cost functions in $\mathbb{R}^n$, then move to a general Euclidean space and finally to the more general case of a Riemannian manifold.

The trust-region method (see [27, 102]) is an iterative method for minimizing a cost function. At each iteration step a quadratic model of the cost function is obtained. This model is assumed to be adequate in a region (the trust-region) around the current iterate. Then, an update is computed as the minimizer of the model in the trust region. The quality of the updated iterate is evaluated; it is consequently accepted or rejected and the trust-region radius is adjusted.

6.1.1 Basic trust-region method

In $\mathbb{R}^n$, given $f : \mathbb{R}^n \to \mathbb{R}$, the trust-region subproblem for finding the update $\eta \in \mathbb{R}^n$ for the current iterate $\mathbf{x} \in \mathbb{R}^n$ is given by

$$\min_{\eta \in \mathbb{R}^n} m(\eta), \quad m(\eta) = f(\mathbf{x}) + \partial f(\mathbf{x})\eta + \frac{1}{2}\eta^T \partial^2 f(\mathbf{x})\eta, \quad \|\eta\| \leq \Delta,$$

where $\partial f(\mathbf{x})$ is the gradient row vector of the first partial derivatives of $f(\mathbf{x})$, $\partial^2 f(\mathbf{x})$ is the Hessian matrix of the second partial derivatives of $f(\mathbf{x})$, and Δ is the trust-region radius. The quality of the model m is evaluated by means of the quotient

$$\rho = \frac{f(\mathbf{x}) - f(\mathbf{x}+\eta)}{m(\mathbf{0}) - m(\eta)}. \tag{6.1}$$

If ρ is close to 0 or negative, then the model is very inaccurate, i.e., the step must be rejected and the trust-region radius must be reduced. If ρ is larger but still small, the step is accepted and the trust-region radius is reduced. Finally, if ρ is close to 1, then there is a good correspondence between the model and the cost function, the step is accepted and the trust-region radius can be increased.

In a Euclidean space E, if $f : E \to \mathbb{R}$ and $\langle \cdot, \cdot \rangle$ is the inner product in E, then the trust-region subproblem can be formulated as follows without resorting to a basis of E :

$$\begin{aligned} \min_{\eta \in E} m(\eta), \quad m(\eta) &= f(\mathbf{x}) + \mathrm{D}f(\mathbf{x})[\eta] + \tfrac{1}{2}\mathrm{D}^2 f(\mathbf{x})[\eta, \eta] \\ &= f(\mathbf{x}) + \langle \operatorname{grad} f(\mathbf{x}), \eta \rangle + \tfrac{1}{2}\langle \operatorname{Hess} f(\mathbf{x})[\eta], \eta \rangle, \quad \langle \eta, \eta \rangle \leq \Delta_{\mathbf{x}}^2, \end{aligned} \tag{6.2}$$

where $\mathrm{D}F(\mathbf{x})[\mathbf{z}]$ denotes the directional derivative of the function F at $\mathbf{x}$ in the direction of $\mathbf{z}$; $\operatorname{grad} f(\mathbf{x})$ and $\operatorname{Hess} f(\mathbf{x})$ stand for the gradient and the Hessian of f at $\mathbf{x}$.

6.1.2 Trust-region method on a Riemannian manifold

Let now the function f be defined on a Riemannian manifold M. In this case, the trust-region subproblem at a point $\mathbf{x} \in M$ takes place on the tangent space $T_{\mathbf{x}}M$, which is a Euclidean space. The update vector $\eta \in T_{\mathbf{x}}M$ is then a tangent vector. Its direction corresponds to the direction in which the next iterate is to be found and its length indicates how far in this direction to go. However, the new iterate should be on the manifold and not on the tangent space. The actual correspondence between η and the new point on the manifold is given through a retraction (Section 3.6.1).

Given a cost function $f : M \to \mathbb{R}$ and a current iterate $\mathbf{x}_k \in M$ we look for $\mathbf{x}^* \in M$, such that $f(\mathbf{x}^*)$ is minimal [1]. Using $R_{\mathbf{x}_k}$, the minimization problem for f on M is locally mapped onto a minimization problem on $T_{\mathbf{x}_k}M$ for the pullback $\hat{f}_{\mathbf{x}_k} : T_{\mathbf{x}_k}M \to \mathbb{R}$ of f,

$$\hat{f}_{\mathbf{x}_k} : \xi \mapsto f(R_{\mathbf{x}_k}(\xi)).$$

Following (6.2), the trust-region subproblem for $T_{\mathbf{x}_k}M$ becomes

$$\begin{aligned} \min_{\eta \in T_{\mathbf{x}_k}M} m_{\mathbf{x}_k}(\eta), \; m_{\mathbf{x}_k}(\eta) &= \hat{f}_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k}) + \mathrm{D}\hat{f}_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k})[\eta] + \tfrac{1}{2}\mathrm{D}^2 \hat{f}_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k})[\eta, \eta] \\ &= \hat{f}_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k}) + \langle \operatorname{grad} \hat{f}_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k}), \eta \rangle + \tfrac{1}{2}\langle \operatorname{Hess} \hat{f}_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k})[\eta], \eta \rangle \\ &= f(\mathbf{x}_k) + \langle \operatorname{grad} f(\mathbf{x}_k), \eta \rangle + \tfrac{1}{2}\langle \operatorname{Hess} f(\mathbf{x}_k)[\eta], \eta \rangle, \end{aligned} \tag{6.3}$$

where $\langle \eta, \eta \rangle \leq \Delta_k^2$ and $\operatorname{Hess} f(\mathbf{x}_k)$ is the Riemannian Hessian [5]. An approximate but sufficiently accurate solution to the trust-region subproblem is given by the truncated conjugate gradient algorithm (tCG) as suggested in [127, 132]. An advantage of this algorithm is that the Hessian matrix is not required explicitly but only its application to a tangent vector has to be computed. Other possible methods for (approximately) solving

the trust-region subproblem (6.3) can be found in [97, 27]. The candidate for the new iterate is given by

$$\mathbf{x}_+ = R_{\mathbf{x}_k}(\eta_k),$$

where η_k is the solution of (6.3). The quotient

$$\rho_k = \frac{f(\mathbf{x}_k) - f(R_{\mathbf{x}_k}(\eta_k))}{m_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k}) - m_{\mathbf{x}_k}(\eta_k)} = \frac{\hat{f}_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k}) - \hat{f}_{\mathbf{x}_k}(\eta_k)}{m_{\mathbf{x}_k}(\mathbf{0}_{\mathbf{x}_k}) - m_{\mathbf{x}_k}(\eta_k)} \tag{6.4}$$

is evaluated in the same way as (6.1), the new iterate is accepted or rejected and the trust-region radius is updated.

Under mild conditions, the trust-region algorithm converges globally (i.e., for all initial points) to stationary points [1]. Furthermore, since the cost function is monotonically decreasing, it is expected that convergence to a local minimum is achieved in practice. Convergence to a saddle point or a local maximum is only observed in specially designed numerical examples.

6.2 Riemannian trust-region based rank-(R_1, R_2, R_3) approximation

In this section, we apply the Riemannian trust-region scheme described in Section 6.1 to the problem (2.3). For this purpose, we need to go through the "checklist" in [1, §5.1] and give closed-form expressions for all the necessary components. Due to the invariance property (3.1) of $\overline{g}$ we consider a Riemannian quotient manifold on which the computations take place. For the inner product on the manifold and the retraction function, we recall expressions from Section 3.4 and Section 3.6.1 and generalize them to our product manifold. Then we present closed-form expressions for the gradient and Hessian of the cost function, as required by [1, §5.1]. Note that in order to maximize $\overline{g}$ using the trust-region scheme, we, in fact, minimize the function $-\overline{g}$.

We choose the Grassmann manifold (Section 3.4) as a working space but since in (2.3) we optimize three matrices instead of one, we need to deal with a manifold that is a bit more complicated , namely the product manifold

$$M = St(R_1, I_1)/O_{R_1} \times St(R_2, I_2)/O_{R_2} \times St(R_3, I_3)/O_{R_3}. \tag{6.5}$$

This manifold can be thought of as a product of three Grassmann manifolds. The function $\overline{g}$ has one and only one projection $g : M \to \mathbb{R}$,

$$g : (\mathbf{U}O_{R_1}, \mathbf{V}O_{R_2}, \mathbf{W}O_{R_3}) \mapsto \overline{g}(\mathbf{U}, \mathbf{V}, \mathbf{W}). \tag{6.6}$$

As explained in Chapter 3, the elements of a quotient manifold are more difficult to comprehend because they are equivalence classes of matrices, but simplifications in the

formulas and better convergence results follow, which justify (6.6). It will also be useful to define an extension of $\bar{g}$, $\tilde{g} : \mathbb{R}^{I_1\times R_1} \times \mathbb{R}^{I_2\times R_2} \times \mathbb{R}^{I_3\times R_3} \to \mathbb{R}$,

$$\tilde{g} : (\mathbf{U},\mathbf{V},\mathbf{W}) \mapsto \|\mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T\|^2, \tag{6.7}$$

the domain of which is a vector space, so that all classical techniques of real analysis are applicable. Another advantage of the latter function is that it is suitable for computer computations since matrices are readily implemented as two-dimensional arrays.

Even though we can work with $\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})$ as a representation of $g(\mathbf{U},\mathbf{V},\mathbf{W})$, we still have to keep in mind the structure of M. For this purpose, some additional notions have to be taken into account (see [5] for details). The main ideas were explained in Chapter 3. Here, we modify the formulas to correspond to the product of Grassmann manifolds M from (6.5).

The inner product on the horizontal space $\mathscr{H}_{(\mathbf{U},\mathbf{V},\mathbf{W})}$ is defined as the sum of the three inner products of the corresponding components, i.e.,

$$\begin{aligned}\langle(\mathbf{X}_1^{(1)},\mathbf{X}_1^{(2)},\mathbf{X}_1^{(3)}),(\mathbf{X}_2^{(1)},\mathbf{X}_2^{(2)},\mathbf{X}_2^{(3)})\rangle \\ = \operatorname{trace}(\mathbf{X}_1^{(1)T}\mathbf{X}_2^{(1)}) + \operatorname{trace}(\mathbf{X}_1^{(2)T}\mathbf{X}_2^{(2)}) + \operatorname{trace}(\mathbf{X}_1^{(3)T}\mathbf{X}_2^{(3)}).\end{aligned} \tag{6.8}$$

We will apply the following orthogonal projection

$$P_{(\mathbf{U},\mathbf{V},\mathbf{W})}(\mathbf{Z}_\mathbf{U},\mathbf{Z}_\mathbf{V},\mathbf{Z}_\mathbf{W}) = ((\mathbf{I}-\mathbf{U}\mathbf{U}^T)\mathbf{Z}_\mathbf{U},(\mathbf{I}-\mathbf{V}\mathbf{V}^T)\mathbf{Z}_\mathbf{V},(\mathbf{I}-\mathbf{W}\mathbf{W}^T)\mathbf{Z}_\mathbf{W}). \tag{6.9}$$

It projects the columns of $\mathbf{Z}_\mathbf{U},\mathbf{Z}_\mathbf{V}$ and $\mathbf{Z}_\mathbf{W}$ onto the orthogonal complements of the column spaces of $\mathbf{U},\mathbf{V}$ and $\mathbf{W}$, respectively.

For the retraction R of a given tangent vector $(\mathbf{Z}_\mathbf{U},\mathbf{Z}_\mathbf{V},\mathbf{Z}_\mathbf{W})$ at a point $(\mathbf{U},\mathbf{V},\mathbf{W})$, we use the following definition:

$$R_{(\mathbf{U},\mathbf{V},\mathbf{W})}(\mathbf{Z}_\mathbf{U},\mathbf{Z}_\mathbf{V},\mathbf{Z}_\mathbf{W}) = (\operatorname{qf}(\mathbf{U}+\mathbf{Z}_\mathbf{U}),\operatorname{qf}(\mathbf{V}+\mathbf{Z}_\mathbf{V}),\operatorname{qf}(\mathbf{W}+\mathbf{Z}_\mathbf{W})), \tag{6.10}$$

where qf denotes the $\mathbf{Q}$ factor of the thin $\mathbf{QR}$ decomposition [57, §5.2]. As mentioned earlier, other possibilities exist for the retraction function. However, our choice is computationally efficient and motivated by the fact that we are only interested in column spaces of the matrices $\mathbf{U},\mathbf{V}$ and $\mathbf{W}$ and not in their actual values.

Finally, closed-form expressions for the gradient $\operatorname{grad} g$, and the Hessian $\operatorname{Hess} g$ are obtained in the next two sections.

6.2.1 Computation of the gradient

We have the following link between the *gradient* of g and the gradient of $\tilde{g}$ [5, (3.39)]

$$\operatorname{grad} g(\mathbf{U},\mathbf{V},\mathbf{W}) = P_{(\mathbf{U},\mathbf{V},\mathbf{W})} \operatorname{grad} \tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W}), \tag{6.11}$$

where $P_{(\mathbf{U},\mathbf{V},\mathbf{W})}(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})$ is the orthogonal projection (6.9). Next, based on (2.1), we derive the differential of $\tilde{g}$ as

$$\begin{aligned} \mathrm{D}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] &= 2\operatorname{trace}(\mathbf{Z_U}^T \mathbf{A}_{(1)} (\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T (\mathbf{A}_{(1)})^T \mathbf{U}) \\ &\quad +2\operatorname{trace}(\mathbf{Z_V}^T \mathbf{A}_{(2)} (\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T (\mathbf{A}_{(2)})^T \mathbf{V}) \\ &\quad +2\operatorname{trace}(\mathbf{Z_W}^T \mathbf{A}_{(3)} (\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T (\mathbf{A}_{(3)})^T \mathbf{W}). \end{aligned}$$

The formula for the gradient of $\tilde{g}$ follows directly from the latter expression, namely

$$\begin{aligned} \operatorname{grad} \tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W}) &= (\operatorname{grad}_{\mathbf{U}} \tilde{g}, \operatorname{grad}_{\mathbf{V}} \tilde{g}, \operatorname{grad}_{\mathbf{W}} \tilde{g}) \\ &= (2\mathbf{A}_{(1)} (\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T (\mathbf{A}_{(1)})^T \mathbf{U}, \\ &\quad 2\mathbf{A}_{(2)} (\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T (\mathbf{A}_{(2)})^T \mathbf{V}, \\ &\quad 2\mathbf{A}_{(3)} (\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T (\mathbf{A}_{(3)})^T \mathbf{W}). \end{aligned} \tag{6.12}$$

Combining (6.11), (6.9) and (6.12), a closed-form expression for the gradient of g is obtained.

6.2.2 Computation of the Hessian

A starting point for computing the *Hessian* of g is the following formula [5, (7.2) and Prop. 5.3.4]

$$\operatorname{Hess} g(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] = P_{(\mathbf{U},\mathbf{V},\mathbf{W})} \mathrm{D}(\operatorname{grad} g)(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]. \tag{6.13}$$

Taking into account equations (6.11) and (6.9), we can transform the first component of $\mathrm{D}(\operatorname{grad} g)(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]$ to

$$\begin{aligned} &\mathrm{D}((\mathbf{I}-\mathbf{U}\mathbf{U}^T)\operatorname{grad}_{\mathbf{U}} \tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \\ &= \mathrm{D}(\operatorname{grad}_{\mathbf{U}} \tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathrm{D}(\mathbf{U}\mathbf{U}^T \operatorname{grad}_{\mathbf{U}} \tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \\ &= \mathrm{D}(\operatorname{grad}_{\mathbf{U}} \tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_U}\mathbf{U}^T \operatorname{grad}_{\mathbf{U}} \tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W}) \\ &\quad -\mathbf{U}\mathrm{D}\{\mathbf{U}^T \operatorname{grad}_{\mathbf{U}} \tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\}[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]. \end{aligned}$$

Similar expressions are obtained for the other two components. Substituting these expressions in (6.13), using (6.9) and the equality $(\mathbf{I}-\mathbf{U}\mathbf{U}^T)\mathbf{U}=\mathbf{0}$, (6.13) is simplified to

$$\begin{aligned}
&\text{Hess } g(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] = \\
&\Big((\mathbf{I}-\mathbf{U}\mathbf{U}^T)\Big(\mathrm{D}(\mathrm{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_U}\mathbf{U}^T\mathrm{grad}_{\mathbf{U}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\Big), \\
&\quad(\mathbf{I}-\mathbf{V}\mathbf{V}^T)\Big(\mathrm{D}(\mathrm{grad}_{\mathbf{V}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_V}\mathbf{V}^T\mathrm{grad}_{\mathbf{V}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\Big), \\
&(\mathbf{I}-\mathbf{W}\mathbf{W}^T)\Big(\mathrm{D}(\mathrm{grad}_{\mathbf{W}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_W}\mathbf{W}^T\mathrm{grad}_{\mathbf{W}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\Big)\Big).
\end{aligned} \tag{6.14}$$

The only quantities in (6.14) that still lack a closed-form expression are of the form $\mathrm{D}(\mathrm{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]$. From (6.12) we have:

$$\begin{aligned}
\mathrm{D}(\mathrm{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \;=\; & 2\mathbf{A}_{(1)}(\mathbf{Z_V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U} \\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{Z_W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U} \\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{Z_V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U} \\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{Z_W})^T(\mathbf{A}_{(1)})^T\mathbf{U} \\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{Z_U}\,.
\end{aligned}$$

By analogy,

$$\begin{aligned}
\mathrm{D}(\mathrm{grad}_{\mathbf{V}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \;=\; & 2\mathbf{A}_{(2)}(\mathbf{Z_W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V} \\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{Z_U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V} \\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{Z_W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V} \\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{Z_U})^T(\mathbf{A}_{(2)})^T\mathbf{V} \\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{Z_V}
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{D}(\mathrm{grad}_{\mathbf{W}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \;=\; & 2\mathbf{A}_{(3)}(\mathbf{Z_U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{W} \\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{Z_V})(\mathbf{U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{W} \\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{Z_U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{W} \\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{Z_V})^T(\mathbf{A}_{(3)})^T\mathbf{W} \\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{Z_W}\,.
\end{aligned}$$

A closed-form expression for Hess $g(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]$ is derived by substituting the last three expressions and (6.12) in (6.14).

6.2.3 Summary of the algorithm

In this section, we summarize the whole algorithm as Algorithm 4. In order to simplify the presentation, let $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$ and $\mathbf{Z} = (\mathbf{Z}_\mathbf{U}, \mathbf{Z}_\mathbf{V}, \mathbf{Z}_\mathbf{W})$.

Algorithm 4 Trust-region based algorithm

1: Obtain initial iterate $\mathbf{X}_0 \in M$, e.g., using the truncated HOSVD from Section 2.4.1 and set initial value for Δ_0.
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: Acquire a solution $\mathbf{Z}_k$ of the trust-region subproblem

$$\min_{\mathbf{Z} \in T_{\mathbf{X}_k} M} m_{\mathbf{X}_k}(\mathbf{Z}), \quad \text{subject to } \langle \mathbf{Z}, \mathbf{Z} \rangle \leq \Delta_k^2,$$

where

$$m_{\mathbf{X}_k}(\mathbf{Z}) = -g(\mathbf{X}_k) + \langle -\operatorname{grad} g(\mathbf{X}_k), \mathbf{Z} \rangle + \frac{1}{2} \langle -\operatorname{Hess} g(\mathbf{X}_k)[\mathbf{Z}], \mathbf{Z} \rangle.$$

An approximate solution is given by the tCG algorithm [127, 1].
4: Form the quotient

$$\rho_k = \frac{-g(\mathbf{X}_k) + g(R_{\mathbf{X}_k}(\mathbf{Z}_k))}{m_{\mathbf{X}_k}(\mathbf{0}_{\mathbf{X}_k}) - m_{\mathbf{X}_k}(\mathbf{Z}_k)}.$$

$$\mathbf{X}_{k+1} = \begin{cases} R_{\mathbf{X}_k}(\mathbf{Z}_k), & \text{if } \rho_k \text{ is large} \\ \mathbf{X}_k, & \text{otherwise.} \end{cases}$$

$$\Delta_{k+1} = \begin{cases} \frac{1}{4}\Delta_k, & \text{if } \rho_k \text{ is small} \\ 2\Delta_k, & \text{if } \rho_k \text{ is large} \\ \Delta_k, & \text{otherwise.} \end{cases}$$

5: **end for**
6: $\mathbf{X} = \mathbf{X}_{k+1}$.
7: Compute

$$\mathscr{B} = \mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T,$$

where $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$. The best low multilinear rank approximation $\hat{\mathscr{A}}$ is given by

$$\hat{\mathscr{A}} = \mathscr{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}.$$

6.3 Comparison with HOOI

In this section, we perform numerical experiments illustrating the properties of the proposed algorithm. All experiments are carried out in parallel for the trust-region and HOOI algorithms. The simulations were run using MATLAB. The machine precision is approximately $2 * 10^{-16}$.

6.3.1 Convergence speed

We start our comparison with the convergence speed of both algorithms. HOOI ([37], see also Section 2.4.2) only has linear convergence, whereas the local rate of convergence of the trust-region based algorithm is superlinear [1]. This improvement in the trust-region based algorithm is due to using second order information of the cost function. For illustration, we considered tensors $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ with elements uniformly distributed over the interval $[0,1]$. We set $R_1 = R_2 = R_3 = 5$. Initial matrices were taken from the truncated HOSVD. Then both algorithms were run until convergence ($\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) or until a maximum of 100 iterations was reached. In Figure 6.1, the results of this experiment are shown for one representative example. Three stages of the trust-region algorithm evolution can be distinguished. Due to a large initial trust-region radius, a poor correspondence between the cost function and the quadratic

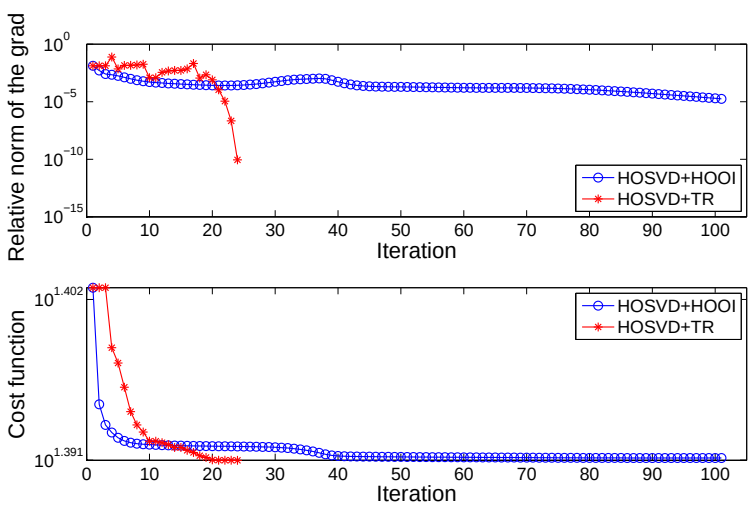
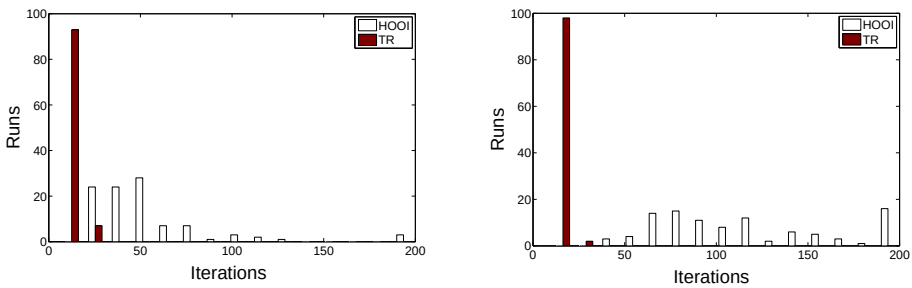


Figure 6.1: *Evolution of the relative norm of the gradient* $\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X})$ *and the cost function (2.2). The tensor* $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ *has elements uniformly distributed over the interval* $[0,1]$*;* $R_1 = R_2 = R_3 = 5$*. The starting values are taken from the truncated HOSVD.*

model was obtained for the first two updates and as a result they were rejected. In the plots this corresponds to constant values of the cost function and of the relative norm of the gradient. After reducing the trust-region radius, the values of the quotient (6.4) considerably improved, so the majority of the following steps were successful. As a consequence, during this second phase, the value of the cost function decreased monotonically as it can be seen from the bottom plot. The relative norm of the gradient did not decrease monotonically because of the complex nature of the cost function. However, its decrease is not required in order to improve the cost function value. Finally, in the last steps, the iterates were close enough to the solution for superlinear convergence, as can be observed in the gradient plot.

The number of iterations necessary for the algorithms to converge is strongly related to the convergence speed. In Figure 6.2, we compare the two algorithms for tensors



(a) $(R_1, R_2, R_3) = (7, 8, 9)$ (b) $(R_1, R_2, R_3) = (2, 2, 2)$

Figure 6.2: *Number of iterations for HOOI and trust-region algorithms for tensors* $\mathscr{A} \in \mathbb{R}^{10\times 10\times 10}$ *with elements taken from a normal distribution with zero mean and unit standard deviation. Performed are 100 runs with initial values* $\mathbf{U}_0, \mathbf{V}_0$, *and* $\mathbf{W}_0$ *taken from the truncated HOSVD. A run was stopped if the corresponding algorithm did not converge in 200 iterations* ($\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$).

$\mathscr{A} \in \mathbb{R}^{10\times 10\times 10}$ with elements taken from a normal distribution with zero mean and unit standard deviation and for two multilinear ranks, $(7, 8, 9)$ and $(2, 2, 2)$, respectively. The results shown are for 100 runs and initial values were taken from the truncated HOSVD. Both algorithms were run until convergence ($\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) or until a maximum of 200 iterations was reached. The trust-region algorithm did not reach the maximum number of iterations in either test. HOOI had 3 runs in (a) and 13 runs in (b) with 200 iterations. It can be concluded that in order to converge, the trust-region based algorithm requires less iterations than HOOI. The difference seems to be stronger for a smaller multilinear rank.

6.3.2 Computational cost

While the trust-region method converges faster than HOOI, the cost for one iteration is higher than the cost for one HOOI iteration, which is $O(I^3R + IR^4 + R^6)$ as shown in Section 2.4.2 [51, 72]. For simplicity, we assume that $R_1 = R_2 = R_3 = R$ and $I_1 = I_2 = I_3 = I$.

In order to compute the cost for one trust-region iteration, we need to estimate the cost of one inner tCG iteration and multiply it by the number of tCG iterations within one trust-region step. The cost for the operations outside the inner tCG loop is smaller and can be neglected. The computationally heaviest operation in the trust-region algorithm is the computation of the Hessian, or more precisely, the application of the Hessian on a tangent vector. This is dominated by expressions such as $\mathbf{A}_{(1)}(\mathbf{Z}_\mathbf{V} \otimes \mathbf{W})$. Another representation of these expressions is $(\mathscr{A} \bullet_2 \mathbf{Z}_\mathbf{V}^T) \bullet_3 \mathbf{W}^T = (\mathscr{A} \bullet_3 \mathbf{W}^T) \bullet_2 \mathbf{Z}_\mathbf{V}^T$, which requires $O(I^3R)$ flops. Such expressions are computed in each inner iteration, i.e., in each step of tCG. However, the products $\mathscr{A} \bullet_1 \mathbf{U}^T$, $\mathscr{A} \bullet_2 \mathbf{V}^T$, $\mathscr{A} \bullet_3 \mathbf{W}^T$, $\mathscr{A} \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T$, etc., which involve only the tensor $\mathscr{A}$ and the matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$ and which do not involve the components $\mathbf{Z}_\mathbf{U}, \mathbf{Z}_\mathbf{V}, \mathbf{Z}_\mathbf{W}$ of tangent vectors, can be computed at the beginning of each trust-region step. They require $O(I^3R)$ flops. Thus, within the inner iterations only expressions of the form $(\mathscr{A} \bullet_3 \mathbf{W}^T) \bullet_2 \mathbf{Z}_\mathbf{V}^T$, given $\mathscr{A} \bullet_3 \mathbf{W}^T$, are computed. The cost of such an operation is only $O(I^2R^2)$. Finally, the maximum number of inner iterations per trust-region iteration is $3(I-R)R$, which equals the dimension of the manifold. This leads us to a total cost of $O(I^3R) + 3(I-R)R\,O(I^2R^2) = O(I^3R^3)$ for one trust-region step. It should be taken into account that in applications, the values of R_n are often much smaller than the corresponding values of I_n.

Note, however, that one can reduce the computational cost of the trust-region algorithm without losing its fast local convergence rate. This can be done by choosing a stopping criterion for the inner iteration based on the gradient of the cost function as in [1]. In this case, few inner tCG steps are taken when the outer iterate is far away from the solution, i.e., when the gradient is large, and more inner tCG steps are taken close to the solution. This behavior is illustrated in Figure 6.3. The tCG inner iteration is stopped according to

$$\|r_j\| \leq \|r_0\| \min(\|r_0\|^\theta, \kappa),$$

where r_j is the residual at the j^{th} step of tCG; see [1, §3] for details. We chose $\theta = 1$, which yields quadratic convergence of the trust-region algorithm. We used this inner stopping criterion for all our numerical experiments involving the trust-region method. For Figure 6.3, the entries of $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ are taken from a normal distribution with zero mean and unit standard deviation. We set $R_1 = R_2 = R_3 = 2$ and $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$ are obtained from the truncated HOSVD. The algorithm was stopped when $\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$.

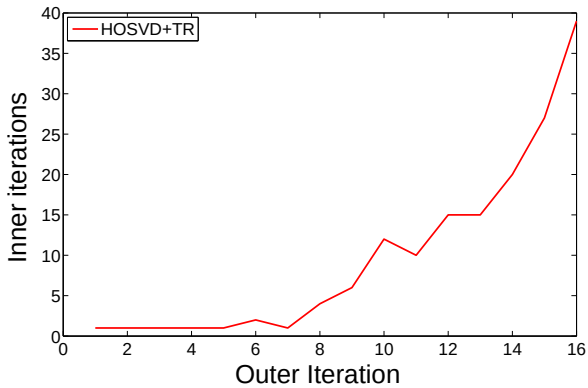


Figure 6.3: *Number of inner tCG steps per trust-region step. The tensor $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ has elements taken from a normal distribution with zero mean and unit standard deviation; $R_1 = R_3 = R_3 = 2$ and $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$ are taken from the truncated HOSVD. Few tCG iterations are performed in the first trust-region steps and more in the neighborhood of the solution.*

6.4 Comparison with Newton-type methods

Besides HOOI, Newton-type methods for the best rank-(R_1, R_2, R_3) approximation have appeared in the literature ([72, 51, 118], see also Chapter 5). These methods have local quadratic convergence rate and their computational cost per iteration is of the same order as the one of the trust-region method. However, they are not globally convergent and depend strongly on the initial values of the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$. In practice, these methods might diverge. The truncated HOSVD often gives good initial values but sometimes these values are not good enough. On the other hand, the trust-region method converges globally to local optima except for very special examples that are artificially constructed.

Another advantage of the trust-region method is that the cost function is monotonically decreasing. This explains why convergence to saddle points or local maxima is not observed in practice. Newton methods do not distinguish between minima, maxima and saddle points. This means that if the stationary points are close to each other, even if a relatively good starting point is chosen, these algorithms might converge to a maximum or to a saddle point instead of to a minimum.

To illustrate the above, we considered tensors $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ with elements uniformly distributed over the interval $[0, 1]$ and estimated their best rank-$(5, 5, 5)$ approximation. We started from values taken from the truncated HOSVD and performed 20 iterations of HOOI. Then we ran HOOI, the geometric Newton [72] and the trust-region algorithms until convergence, with a maximum of 500 iterations. In Figure 6.4, one example is shown where HOOI and the trust-region algorithm both converge to a local minimum of

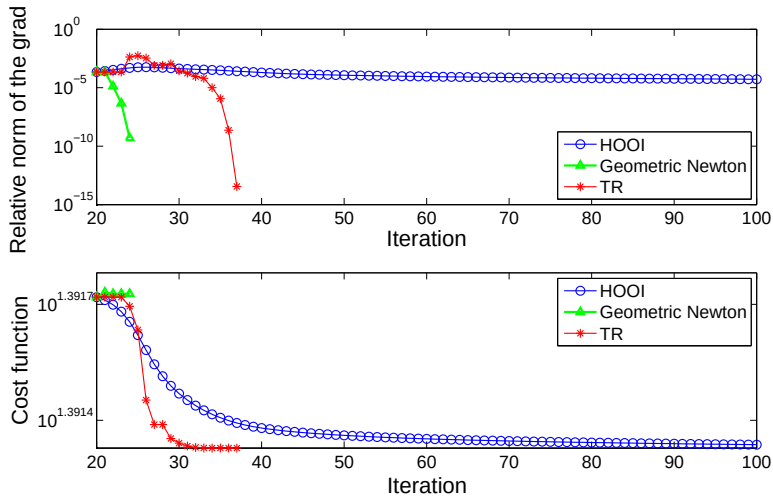


Figure 6.4: *Evolution of the relative norm of the gradient $\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X})$ and the cost function (2.2). The tensor $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ has elements uniformly distributed over the interval $[0,1]$; $R_1 = R_2 = R_3 = 5$. The starting point of the plots is obtained by initializing with matrices taken from the truncated HOSVD and performing 20 additional HOOI iterations.*

(2.2) but the geometric Newton algorithm converges to a saddle point. For clarity, the figure presents only the iterations over the interval $[20,100]$. The type of the end points was evaluated using the Hessian of the cost function. Cases like this seem rather rare. Often, the Newton-type and the trust-region algorithms perform in a similar way in the same setting, i.e., converge to a local minimum with a similar number of iterations. In our simulations, it also happened that both types of algorithms converged to a local minimum but that the Newton-type algorithm needed considerably more iterations. This occurred especially when the starting point of the Newton algorithm had an indefinite Hessian.

In tests with low multilinear rank tensors affected by additive noise with a modest signal-to-noise ratio, the performance of the Newton-type and trust-region algorithms was similar. We considered rank-$(5,5,5)$ tensors $\mathscr{T} \in \mathbb{R}^{20\times 20\times 20}$, affected by additive noise in the following way

$$\mathscr{A} = \mathscr{T}/\|\mathscr{T}\| + 0.1\,\mathscr{E}/\|\mathscr{E}\|, \tag{6.15}$$

in which $\mathscr{E} \in \mathbb{R}^{20\times 20\times 20}$ represents noise with elements uniformly distributed over the interval $[0,1]$. The tensor $\mathscr{T}$ was constructed as the product of a tensor $\mathscr{C} \in \mathbb{R}^{5\times 5\times 5}$ and three matrices $\mathbf{M}^{(i)} \in \mathbb{R}^{20\times 5}$, $i = 1,2,3$, all with elements uniformly distributed over the interval $[0,1]$, i.e.,

$$\mathscr{T} = \mathscr{C} \bullet_1 \mathbf{M}^{(1)} \bullet_2 \mathbf{M}^{(2)} \bullet_3 \mathbf{M}^{(3)}.$$

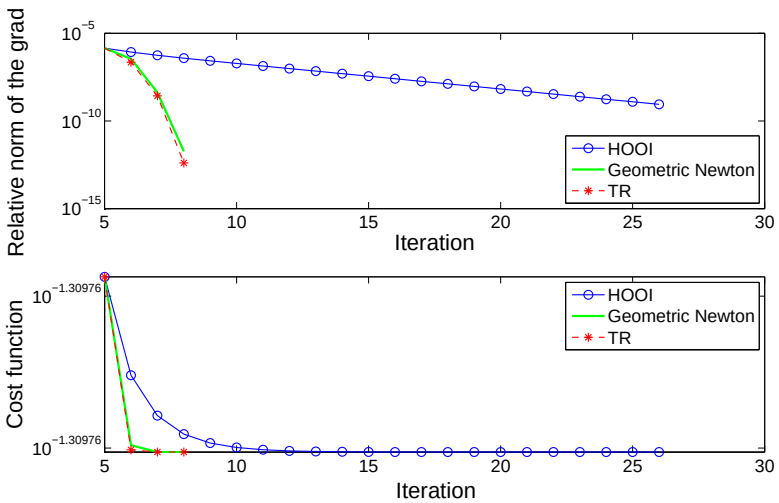


Figure 6.5: *Evolution of the relative norm of the gradient* $\|grad\, g(\mathbf{X})\|/g(\mathbf{X})$ *and the cost function (2.2). The tensor* $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ *is as in (6.15), with* $R_1 = R_2 = R_3 = 5$*. The starting point of the plots is obtained by initializing with matrices taken from the truncated HOSVD and performing 5 additional HOOI iterations.*

In Figure 6.5, one representative example is given. The initial matrices $\mathbf{U}_0$, $\mathbf{V}_0$ and $\mathbf{W}_0$ were taken from the truncated HOSVD and 5 additional HOOI iterations were performed. As it can be seen from the figure, the differential-geometric Newton method and the trust-region algorithm have similar convergence behavior whereas HOOI needs more but computationally less expensive iterations.

We conclude this section by reporting that in our simulations, the Newton methods from [72] and [51] generally behaved in the same way, although for a particular dataset the behavior was sometimes different.

6.5 Conclusions

We have developed a new algorithm for computing the best multilinear rank approximation of higher-order tensors. It is based on the Riemannian trust-region scheme on quotient manifolds. In order to solve the trust-region subproblems, the truncated conjugate gradient method is applied. The algorithm converges globally to local minima of the cost function (2.2). Its convergence rate is superlinear and even quadratic, for a suitable stopping criterion.

In comparison with HOOI, the advantage of the new algorithm is its faster local

convergence rate. The maximal computational cost per iteration of these two algorithms is comparable for small multilinear rank. This is the case in many applications. Moreover, the trust-region based algorithm reaches its maximum number of inner iterations and hence its maximal computational cost per iteration only in iterations close to the solution. Thus, the overall performance of the proposed method is to be preferred in many cases.

In the literature, Newton-type methods have also been presented [72, 51, 118]. Their computational cost per iteration is of the same order as the one of the trust-region method. However, in rather difficult cases, these algorithms have problems distinguishing between minima, maxima and saddle points and might even diverge. The trust-region algorithm overcomes these difficulties.

Chapter 7

Conjugate gradient algorithm

In this chapter, we recall the linear [63] and nonlinear [52, 113] conjugate gradient (CG) methods, also described in [102]. Then, a generalization of nonlinear CG to manifolds is presented [5, 124, 125]. In particular, we consider the implementation of CG on a product of Grassmann manifolds, for the problem of the best low multilinear rank approximation of higher order tensors (2.3) [67, 70].

In Section 7.3, the numerical properties of the proposed CG method are illustrated. We consider the convergence behavior, the computational cost and comment on scheduled restarts and on the precision of the proposed algorithm. A comparison with HOOI (see Section 2.4.2), steepest descent and trust-region (see Chapter 6) algorithms is presented in Section 7.4.

7.1 Conjugate gradient methods

CG methods are iterative methods for minimizing a cost function. At each step a search direction is computed along which the next iterate is to be found. The search direction takes into account all previous directions. A step size is obtained by a line search algorithm and indicates where to take the next iterate along the search direction.

In this section, we summarize the main ideas behind the linear and nonlinear CG methods. The linear CG method is used for solving large linear systems of equations with symmetric positive definite matrices. The nonlinear CG methods are extensions of the linear CG for minimizing cost functions in $\mathbb{R}^n$.

7.1.1 Linear conjugate gradient method

The linear CG method [63] solves large systems of linear equations $\mathbf{Ax} = \mathbf{b}$ having a symmetric positive definite matrix. In practice, the following cost function is minimized instead

$$\phi(\mathbf{x}) = \tfrac{1}{2}\mathbf{x}^T\mathbf{Ax} - \mathbf{x}^T\mathbf{b}\,.$$

The function $\phi(\mathbf{x})$ is convex and quadratic and its minimum coincides with the solution of $\mathbf{Ax} = \mathbf{b}$. The difference $\mathbf{r}_k = \mathbf{b} - \mathbf{Ax}_k$ is called the *residual* of the system at point $\mathbf{x}_k$ and we have $\mathbf{r}_k = \mathbf{b} - \mathbf{Ax}_k = -\text{grad}\ \phi(\mathbf{x}_k)$.

To start CG, an initial point $\mathbf{x}_0$ is chosen and the initial search direction is taken equal to the *steepest descent direction* at $\mathbf{x}_0$, i.e., $\mathbf{p}_0 = \mathbf{r}_0 = -\text{grad}\ \phi(\mathbf{x}_0)$. Every subsequent search direction $\mathbf{p}_{k+1}$, $(k = 0, 1, 2, \ldots)$ of CG is required to be conjugate to all previously generated search directions. It is enough to make sure that $\mathbf{p}_{k+1}$ is *conjugate* to the previous direction $\mathbf{p}_k$, i.e.,

$$\mathbf{p}_{k+1}^T\mathbf{A}\mathbf{p}_k = 0$$

and to compute $\mathbf{p}_{k+1}$ as a linear combination of the steepest descent direction $\mathbf{r}_{k+1} = -\text{grad}\ \phi(\mathbf{x}_{k+1})$ at the current point and $\mathbf{p}_k$. This implies that $\mathbf{p}_{k+1}$ is also conjugate to all $\mathbf{p}_i, i = 0, \ldots, k$ and that all directions are linearly independent. More precisely,

$$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_{k+1}\mathbf{p}_k,$$

where the coefficient β_{k+1} is chosen such that $\mathbf{p}_{k+1}$ and $\mathbf{p}_k$ are conjugate and $\mathbf{p}_{k+1}$ is as close as possible to $\mathbf{r}_{k+1}$. The algorithm is illustrated in Figure 7.1. The step length α_k is

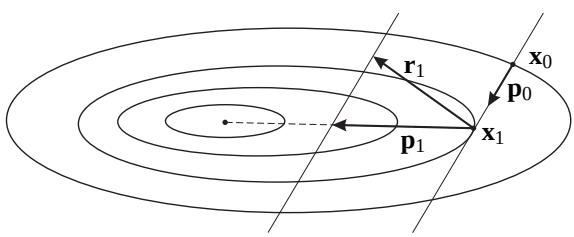


Figure 7.1: *Illustration of the linear CG algorithm.*

chosen as the exact minimizer of ϕ along $\mathbf{p}_k$. If $\mathbf{A} \in \mathbb{R}^{n\times n}$ and the first search direction is $\mathbf{r}_0$, the optimal solution is found in n steps.

The whole procedure is summarized in Algorithm 5.

Algorithm 5 Linear conjugate gradient method

1: Obtain an initial iterate $\mathbf{x}_0$.
2: Set $\mathbf{p}_0 = \mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
3: **for** $k = 0, 1, 2, \ldots$ **do**
4: $\quad \alpha_k = \dfrac{\mathbf{r}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A}\mathbf{p}_k}$,
5: $\quad \mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$,
6: $\quad \beta_{k+1} = -\dfrac{\mathbf{r}_{k+1}^T \mathbf{A}\mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A}\mathbf{p}_k}$,
7: $\quad \mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_{k+1}\mathbf{p}_k$.
8: **end for**

7.1.2 Nonlinear conjugate gradient method

Nonlinear CG methods [52, 113] use the same idea as linear CG but apply it on general nonlinear functions. A few adjustments are necessary. First, a closed-form expression for the optimal step length is not available anymore. A line search algorithm has to be used instead. Second, the residual $\mathbf{r}_k$ should be replaced by $-\text{grad } f(\mathbf{x}_{k+1})$, where f is the cost function to be minimized.

As a consequence, the computation of the coefficient β has to be modified too. There are different strategies, among which the Fletcher-Reeves [52]

$$\beta_{k+1} = \frac{(\text{grad } f(\mathbf{x}_{k+1}))^T \text{ grad } f(\mathbf{x}_{k+1})}{(\text{grad } f(\mathbf{x}_k))^T \text{ grad } f(\mathbf{x}_k)}$$

and Polak-Ribière [113]

$$\beta_{k+1} = \frac{(\text{grad } f(\mathbf{x}_{k+1}))^T (\text{grad } f(\mathbf{x}_{k+1}) - \text{grad } f(\mathbf{x}_k))}{(\text{grad } f(\mathbf{x}_k))^T \text{ grad } f(\mathbf{x}_k)}$$

approach. Other choices for computing β_{k+1} have been presented in the literature [102, §5]. Depending on the choice of step size α and parameter β, different versions of nonlinear CG methods are obtained. All formulas for β_{k+1} reduce to the formula for the same parameter in the linear CG if the function is convex quadratic and if α_k is the exact minimizer along $\mathbf{p}_k$.

Since the cost function is in general not convex quadratic, convergence is obtained after more than n iterations. See Figure 7.2 for an illustration. The convergence properties of nonlinear CG methods are difficult to analyze. Under mild assumptions on the cost function, nonlinear CG converges to stationary points. In practice, it converges to a local minimizer except for some specifically designed examples with particular initial points. In the literature, there are results for combinations of particular line search algorithms and particular choices of β_{k+1}. The issue of global convergence is discussed in [10, 56].

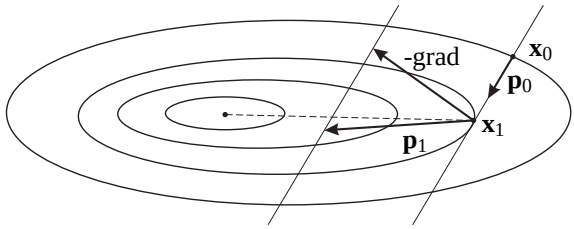


Figure 7.2: *Illustration of the nonlinear CG algorithms.*

In [114], it is argued that the Fletcher-Reeves version can be very inefficient in certain circumstances. The Polak-Ribière version takes a direction closer to the steepest descent direction if the current step is too small. In this way it avoids taking many consecutive small steps. On the other hand, in [116], a counterexample is given where the Polak-Ribière version fails to converge. More convergence results and references can be found in [102, §5].

The advantages of the nonlinear CG methods are their low computational cost and the fact that they do not require a lot of storage space. At each iteration, the cost function and the gradient are evaluated but the computation of the Hessian is not required, as it was the case for the trust-region algorithm from Chapter 6.

7.2 Best rank-(R_1, R_2, R_3) approximation with conjugate gradients

As we did in Chapter 6, we consider as a working space the product of three Grassmann manifolds (6.5), due to the invariance property (3.1) of the cost function (2.3). We first present a general formulation of nonlinear CG in the case of a Riemannian manifold. Then we apply this scheme to the cost function (2.3).

7.2.1 Conjugate gradients on a Riemannian manifold

Nonlinear CG on Riemannian manifolds was first proposed in [124, 125]. This algorithm makes use of the exponential map and parallel translation, which might be inefficient. The algorithm proposed in [5] works with the more general concepts of retraction and vector transport. We refer to Chapter 3 and the references therein for some background information.

In order to generalize the nonlinear CG from functions in $\mathbb{R}^n$ to functions defined on Riemannian manifolds, the expressions for the step length, search direction and the coefficient β_{k+1} have to be adjusted.

In order to guarantee a descent direction at each step, the step size α_k could be taken equal to the exact minimizer of f along η_k. However, this computation could be extremely expensive. In that case, the step size could be computed using a backtracking procedure, searching for an Armijo point. Given scalars $0 < \beta, \sigma < 1$ and $\overline{\alpha} > 0$, this is the point $\eta^A = \beta^m \overline{\alpha} \eta$, where m is the smallest nonnegative integer such that

$$f(\mathbf{x}_k) - f(R_{\mathbf{x}_k}(\beta^m \overline{\alpha} \eta_k)) \geq -\sigma \langle \text{grad } f(\mathbf{x}_k), \beta^m \overline{\alpha} \eta_k \rangle .$$

Here, $\langle \cdot, \cdot \rangle$ is the inner product on the tangent space of the manifold and R is a retraction, see Section 3.6.1. The step size α_k is then taken to be $\alpha_k = \beta^m \overline{\alpha}$. The above condition is called *Armijo condition* or *sufficient decrease* condition. The value σ specifies the meaning of "sufficient". In practice, it is often taken very small. The parameter β controls the grid size (in a logarithmic sense) within the backtracking procedure. The Armijo step size $\beta^m \overline{\alpha}$ is a cheaper alternative of the exact minimizer of f along η_k, preserving the main convergence properties of the CG algorithm.

When computing the new search direction η_{k+1}, another obstacle appears. The formula for η_{k+1} involves the gradient at $\mathbf{x}_{k+1}$ and the previous search direction η_k, which are two vectors in two different tangent spaces. A solution for this problem is to carry η_k over to the tangent space of $\mathbf{x}_{k+1}$. As explained in Section 3.6.2, this is performed by means of a vector transport. The transported η_k is $\mathscr{T}_{\alpha_k \eta_k} \eta_k$, for a well-chosen vector transport $\mathscr{T}$.

Since α_k is not the optimal step size along η_k, it is possible that the new direction is not a descent direction. If this is the case, i.e., if

$$\langle \text{grad } f(\mathbf{x}_{k+1}), \eta_{k+1} \rangle > 0 ,$$

we set $\eta_{k+1} = -\text{grad } f(\mathbf{x}_{k+1})$, which is the steepest descent direction.

Finally, we need to generalize the expression for β_{k+1}. The two possibilities mentioned in Section 7.1.2 are generalized to the case of manifolds as follows: the Fletcher-Reeves formula becomes

$$\beta_{k+1} = \frac{\langle \text{grad } f(\mathbf{x}_{k+1}), \text{grad } f(\mathbf{x}_{k+1}) \rangle}{\langle \text{grad } f(\mathbf{x}_k), \text{grad } f(\mathbf{x}_k) \rangle}$$

and for the generalization of Polak-Ribière we have

$$\beta_{k+1} = \frac{\langle \text{grad } f(\mathbf{x}_{k+1}), \text{grad } f(\mathbf{x}_{k+1}) - \mathscr{T}_{\alpha_k \eta_k}(\text{grad } f(\mathbf{x}_k)) \rangle}{\langle \text{grad } f(\mathbf{x}_k), \text{grad } f(\mathbf{x}_k) \rangle} .$$

7.2.2 Implementation and summary of the algorithm

As in Chapter 6, we work again on the Grassmann manifold. Recall from Chapter 6 the definitions of the manifold M (6.5) on which the computations take place, the additional functions g (6.6) and $\tilde{g}$ (6.7), the inner product (6.8), the projection P (6.9), and the retraction (6.10). For the computation of the gradient we make use of the derivations in Section 6.2.1. For the vector transport, we use (3.12).

We outline the CG-based procedure for the best low multilinear rank approximation in Algorithm 6. For clarity of the presentation, we define $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$.

Algorithm 6 Conjugate gradient based algorithm

1: Take initial iterate $\mathbf{X}_0 \in M$. A good choice of $\mathbf{X}_0$ is often obtained from the truncated HOSVD.
2: Set $\eta_0 = -\text{grad}\, g(\mathbf{X}_0)$.
3: **for** $k = 0, 1, 2, \ldots$ **do**
4: Compute a step size α_k for the direction η_k. The Armijo step size is a reasonable choice.
5: Set $\mathbf{X}_{k+1} = R_{\mathbf{X}_k}(\alpha_k \eta_k)$.
6: Compute β_{k+1} with the Fletcher-Reeves formula

$$\beta_{k+1} = \frac{\langle \text{grad}\, g(\mathbf{X}_{k+1}), \text{grad}\, g(\mathbf{X}_{k+1}) \rangle}{\langle \text{grad}\, g(\mathbf{X}_k), \text{grad}\, g(\mathbf{X}_k) \rangle}$$

or with the Polak-Ribière formula

$$\beta_{k+1} = \frac{\langle \text{grad}\, g(\mathbf{X}_{k+1}), \text{grad}\, g(\mathbf{X}_{k+1}) - \mathscr{T}_{\alpha_k \eta_k}(\text{grad}\, g(\mathbf{X}_k)) \rangle}{\langle \text{grad}\, g(\mathbf{X}_k), \text{grad}\, g(\mathbf{X}_k) \rangle}.$$

7: Set $\eta_{k+1} = -\text{grad}\, g(\mathbf{X}_{k+1}) + \beta_{k+1} \mathscr{T}_{\alpha_k \eta_k}(\eta_k)$.
If $\langle \text{grad}\, g(\mathbf{X}_{k+1}), \eta_{k+1} \rangle > 0$, set $\eta_{k+1} = -\text{grad}\, g(\mathbf{X}_{k+1})$.
8: **end for**
9: $\mathbf{X} = \mathbf{X}_{k+1}$.
10: Compute

$$\mathscr{B} = \mathscr{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T,$$

where $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$. The best low multilinear rank approximation $\hat{\mathscr{A}}$ is given by

$$\hat{\mathscr{A}} = \mathscr{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}.$$

It is expected that the proposed algorithm has properties similar to those of nonlinear CG although theoretical results are difficult to prove. Smith [124, 125] shows that the implementation using the Polak-Ribière formula leads to d-step quadratic convergence towards nondegenerate local minima. The rate of convergence is in general low but the cost per iteration is extremely low. This is due to the fact that the Hessian does not appear in the computations. Under mild assumptions on the cost function convergence to stationary points is expected. Descent directions are guaranteed if we take the steepest descent direction if the proposed direction is not a descent direction itself. Thus, CG converges to local minima unless very special initial values are started from.

7.3 Numerical properties

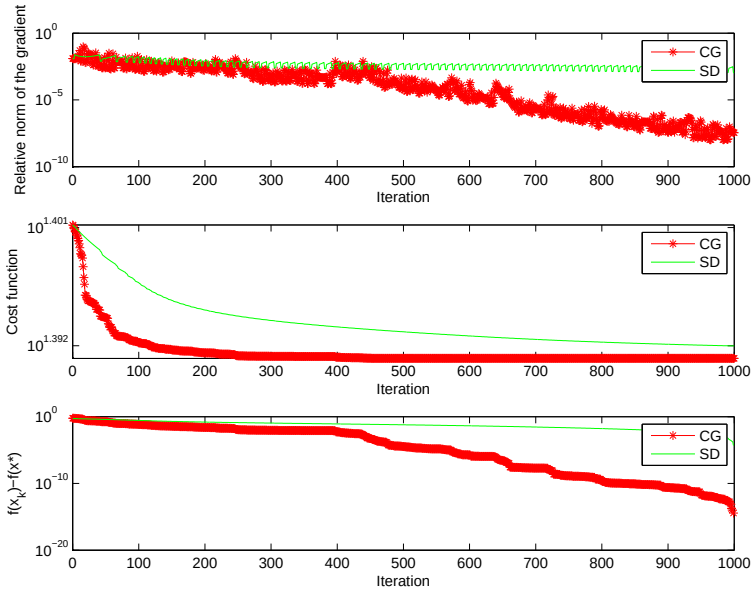
We performed numerical experiments in order to illustrate the properties of the proposed CG algorithm. For our simulations we used MATLAB with machine precision approximately $2 * 10^{-16}$.

We tested Algorithm 6 taking the Armijo step size at each iteration, the Polak-Ribière formula for $\beta_{k+1}, k = 0, 1, 2, \ldots$ and the following parameters: $\sigma = 0.1, \beta = 0.5, \bar{\alpha} = 1$ (initial value). The value of σ was taken small so that points were accepted relatively easily. There is no point in spending too much time on backtracking in one iteration. At the same time, β should not be too close to 1 or to 0 because it is not useful to make the steps within the backtracking procedure too small or too big. We started with an initial value of $\bar{\alpha}$ equal to 1. At each next step we took $\bar{\alpha} = t/\beta$, where t was the previous step size, i.e., we tried again the last unsuccessful step size from the previous CG iteration. We also performed tests with other values of these parameters within some reasonable intervals. The behavior of the algorithm in these tests was similar.

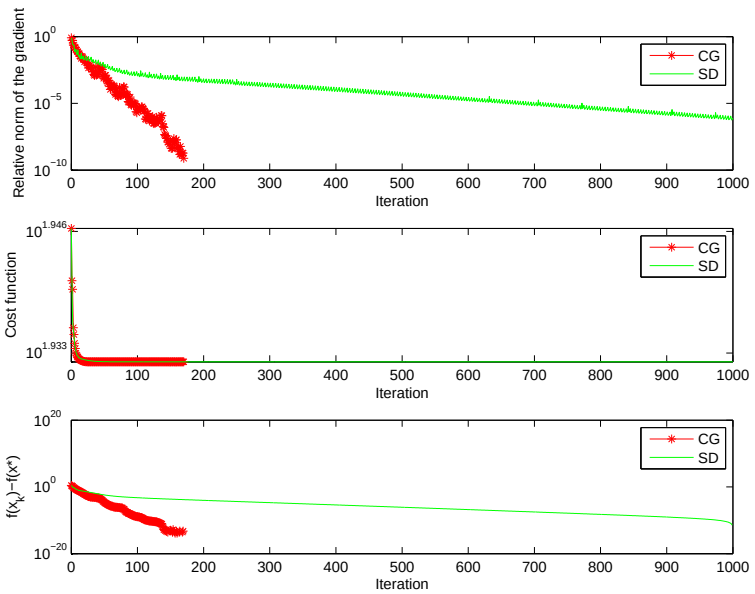
7.3.1 General performance of CG

We considered tensors $\mathscr{A} \in \mathbb{R}^{20 \times 20 \times 20}$ and computed their best rank-$(5,5,5)$ approximations. The starting values were taken from the truncated HOSVD. The tests were run until $\|\text{grad } g(\mathbf{X})\| / g(\mathbf{X}) \leq 10^{-9}$ or until 1000 iterations were reached.

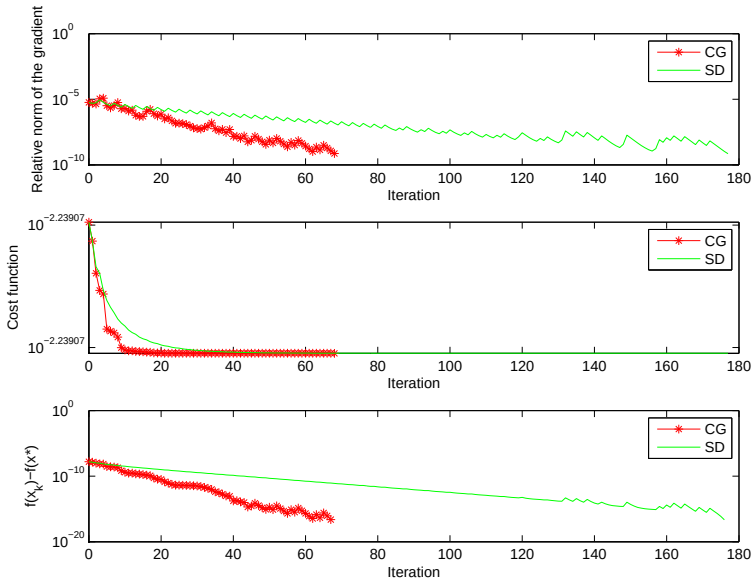
The performance of the proposed CG algorithm applied to different types of tensors was very different. For tensors with elements taken from the uniform distribution over the interval $[0,1]$, the algorithm needed many iterations and often had not converged within 1000 iterations. In Figure 7.3(a), the evolution of the relative norm of the gradient, the original cost function itself and the convergence rate are illustrated. The algorithm performed much better on tensors with elements taken from a normal distribution with zero mean and unit standard deviation as shown in Figure 7.3(b). We also performed tests with tensors that had low multilinear rank and were affected by additive noise. First,



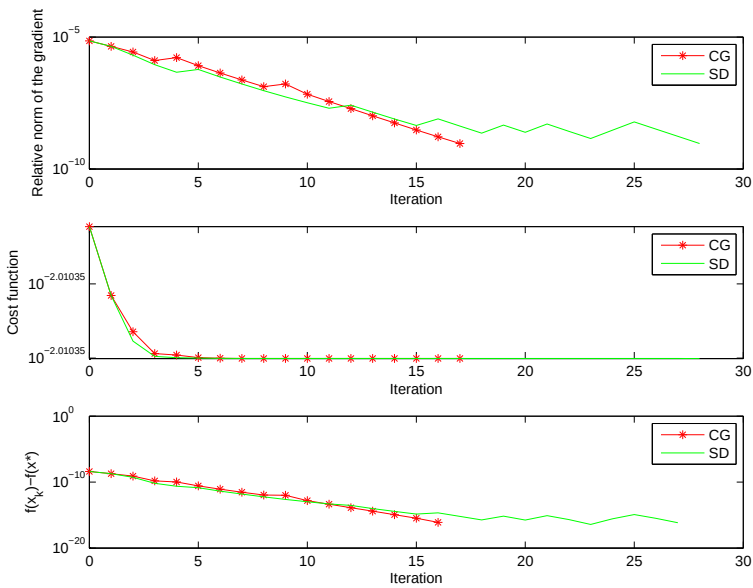
(a) Uniform distribution



(b) Normal distribution



(c) Uniform distribution, $\mathscr{A}$ as in (7.1)



(d) Normal distribution, $\mathscr{A}$ as in (7.1)

Figure 7.3: *Relative norm of the gradient* $\|grad\, g(\mathbf{X})\|/g(\mathbf{X})$ *and cost function (2.2) for the conjugate gradient algorithm (CG) and steepest descent (SD);* $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$*; the multilinear rank of the approximation is* $(5,5,5)$*. The initial points are obtained from the truncated HOSVD.*

a tensor $\mathscr{T}$ was constructed as a product of a tensor $\mathscr{C} \in \mathbb{R}^{5\times5\times5}$ and three matrices $\mathbf{M}^{(i)} \in \mathbb{R}^{20\times5}$, $i = 1,2,3$:

$$\mathscr{T} = \mathscr{C} \bullet_1 \mathbf{M}^{(1)} \bullet_2 \mathbf{M}^{(2)} \bullet_3 \mathbf{M}^{(3)}.$$

The tensor to be approximated was then constructed as

$$\mathscr{A} = \mathscr{T}/\|\mathscr{T}\| + 0.01\,\mathscr{E}/\|\mathscr{E}\|. \tag{7.1}$$

The elements of the tensors $\mathscr{C}$ and $\mathscr{E}$ were taken either from the uniform distribution over $[0,1]$ (Figure 7.3(c)) or from a normal distribution with zero mean and unit standard deviation (Figure 7.3(d)). The matrices $\mathbf{M}^{(i)}$ were random column-wise orthonormal. Namely, they were equal to the **Q** factors of the thin **QR** factorization of matrices with elements drawn from a uniform distribution over $[0,1]$ (Figure 7.3(c)) or from a normal distribution with zero mean and unit standard deviation (Figure 7.3(d)). For tensors in the form of (7.1), CG converged faster than for random tensors. In Figure 7.3, we also illustrate the behavior of the steepest-descent (SD) method (where β_{k+1} from the CG method was set to zero at each iteration) since both CG and SD rely on the same basic tools (gradients and line search). As it can be seen from the plots, CG performs considerably better than SD. In our tests, SD was only competitive in some examples where the tensors $\mathscr{A}$ were as in (7.1) and the considered distribution was the normal distribution. In these examples, both CG and SD converged for a similar number of iterations.

We can conclude that the performance of CG strongly depends on the problem. Namely, in difficult cases, where the tensor does not have a well-conditioned dominant part with low multilinear rank, CG performs less well. The difficulty of the problem is related to the distribution of the mode-n singular values $\sigma_i^{(n)}$ ($n = 1,2,3$; $i = 1,\ldots,I_1/I_2/I_3$) of the original tensor. If, in the first mode, $\sigma_{R_1}^{(1)}/\sigma_{R_1+1}^{(1)}$ is large and if additionally $\sigma_1^{(1)}/\sigma_{R_1}^{(1)}$ is close to 1 and in the other modes we have a similar property, then the problem is well-conditioned. This is the case when the tensor is as in (7.1) and the considered distribution is the normal distribution (see also Figure 7.3(d)). If in (7.1) the uniform distribution over the interval $[0,1]$ is taken (Figure 7.3(c)), the first quotient is still large whereas the second one is not too close to 1 anymore. For tensors as in Figure 7.3(b), both quotients are close to 1. For tensors as in Figure 7.3(a), $\sigma_1^{(n)}/\sigma_{R_n}^{(n)}$ is larger than 1 but $\sigma_{R_n}^{(n)}/\sigma_{R_n+1}^{(n)}$ is close to 1 and the problem is ill-conditioned.

7.3.2 Computational cost

One iteration of CG is rather cheap. The reason is that at each step one only needs to compute the cost function and its gradient. The Hessian of the given function, which was required by the trust-region algorithm, is not used in Algorithm 6. The most expensive expressions are of the type $\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})$, which can be rewritten as $\mathscr{A} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}$. If

we assume for simplicity that $I_1 = I_2 = I_3 = I$ and $R_1 = R_2 = R_3 = R$, the computation of $\mathcal{A} \bullet_2 \mathbf{V}$ requires $O(I^3R)$ flops. The computation of $(\mathcal{A} \bullet_2 \mathbf{V}) \bullet_3 \mathbf{W}$, given $\mathcal{A} \bullet_2 \mathbf{V}$, costs $O(I^2R^2)$ flops. The whole expression can then be computed in $O(I^3R) + O(I^2R^2) = O(I^3R)$ flops.

Of course, there is also the cost of the backtracking procedure. We fixed the parameters of the algorithm in such a way that the number of backtracking steps was minimal. Moreover, in each backtracking step we only need to compute the cost function. The computation of the gradient is not required unless we use the approximated Armijo condition (see Section 7.3.3) and we are very close to the solution (more precisely, when the iterates are close to each other). We can thus assume that the number of backtracking steps and the related computational cost is relatively low.

7.3.3 Precision and restarts

The precision of CG was discussed in [5, 60]. When the distance between the current iterate and the local minimum is close to the square root of the machine precision, the Armijo condition within the line-search procedure can never be satisfied. This results in CG having maximum precision equal to the square root of the machine precision. To overcome this problem, an approximation of the Armijo condition was proposed in [60]. It can be adapted to optimization on a manifold as follows

$$(2\sigma - 1)\langle \mathscr{T}_{\beta^m \overline{\alpha} \eta_k} \text{grad}\, g(\mathbf{X}_k), \mathscr{T}_{\beta^m \overline{\alpha} \eta_k} \eta_k \rangle \leq \langle \text{grad}\, g(R_{\mathbf{X}_k}(\beta^m \overline{\alpha} \eta_k)), \mathscr{T}_{\beta^m \overline{\alpha} \eta_k} \eta_k \rangle .$$

If this condition is satisfied and if

$$g(R_{\mathbf{X}_k}(\beta^m \overline{\alpha} \eta_k)) \geq g(\mathbf{X}_k) + \varepsilon |g(\mathbf{X}_k)|,$$

for some (small) fixed ε, then the iterates are close to each other and there is at least a small decrease of the cost function (and a small increase of the function g that we want to maximize). One then considers the backtracking as terminated successfully with $\alpha_k = \beta^m \overline{\alpha}$ even if the original Armijo condition is not satisfied. In this way, the precision of CG is increased up to the machine precision.

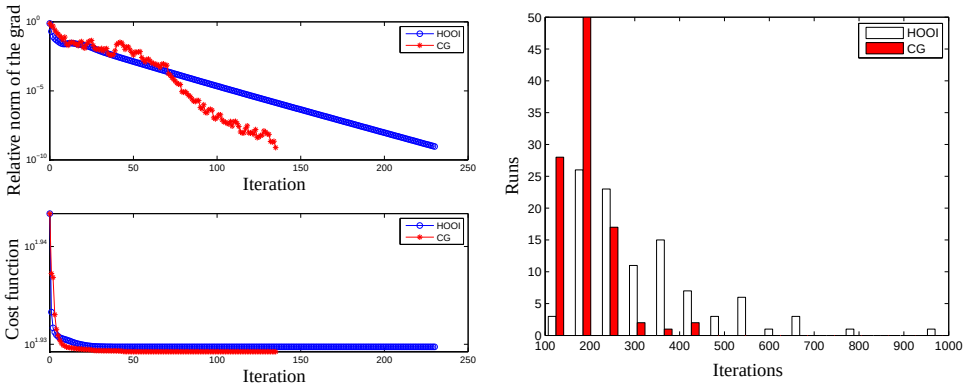
For better convergence results, it is advisable to "restart" the CG algorithm at certain points in order to erase unnecessary old information. A steepest descent direction is taken at the restart points. In practice, this is equivalent to setting the parameter β_{k+1} to zero. Some results on restart procedures are given in [115]. According to one of the best theoretical results about restarts, it is beneficial to restart every d steps, where d is the number of unknown parameters. In this case, the convergence of CG in $\mathbb{R}^n$ is d-step quadratic, which means that quadratic convergence holds for the sequence of iterates obtained by retaining only one step out of d. However, in our problem d is often too large in the sense that the algorithm already converges in less than d iterations. In our examples, $d = 3(I-R)R = 225$.

7.4 Comparison with other algorithms

7.4.1 Comparison with HOOI

HOOI and CG are algorithms with similar theoretical properties. Both algorithms have slow convergence speed but efficient iterations.

The computational cost of one HOOI iteration is of order $O(I^3R+IR^4+R^6)$ (Section 2.4.2, [51, 72]), which is approximately the same as the cost of one CG iteration (see Section 7.3.2). In our numerical experiments, CG converged with a smaller number of iterations in the case of tensors with elements taken from a normal distribution with zero mean and unit standard deviation, see Figure 7.4. For the other three cases mentioned in



(a) Representative example (b) Summary for 100 runs

Figure 7.4: *A representative example and a summary of 100 runs illustrating the number of CG and HOOI iterations. The tensors $\mathcal{A} \in \mathbb{R}^{20\times20\times20}$ are with elements taken from a normal distribution with zero mean and unit standard deviation and $R_1 = R_2 = R_3 = 5$. The initial values were taken from the truncated HOSVD.*

Section 7.3, HOOI converged faster. We mention that the measured computational times of CG and HOOI were also competitive. In general, if the number of CG iterations was smaller than the number of HOOI iterations, also the total computational time for CG was shorter than the total computational time for HOOI and vice-versa. However, we prefer to base our comparison of the computational cost on the number of iterations and the cost per iteration, to avoid differences due to different types of implementation.

7.4.2 Comparison with the trust-region algorithm

In this section, we compare the CG method with the trust-region method, which are very different. CG has low local convergence speed and efficient iterations, whereas the trust-region algorithm has fast local convergence rate but more expensive iterations. However, within each trust-region step, there are inner truncated CG (tCG) iterations. The parameters in CG and tCG are computed in a different way, there are backtracking steps in CG within the line search procedure and, on the other hand, the Hessian in tCG is relatively expensive. Nevertheless, it is still reasonable to assume that the cost of one tCG iteration in the trust-region steps is comparable with the cost of one CG iteration. In our simulations, there were a few backtracking steps in CG for the line search each time. The total computational time of trust-region algorithm was usually similar or higher than the computational time of CG.

In our experiments with tensors with elements taken from the uniform distribution over the interval $[0,1]$, the trust-region algorithm converged with a smaller number of tCG iterations than CG. In the case of tensors with elements taken from a normal distribution with zero mean and unit standard deviation, CG normally converged in a similar number of iterations as the number of tCG iterations within the trust-region steps, see Figure 7.5. Although all runs were generated in a similar way, the particular distributions of the mode-

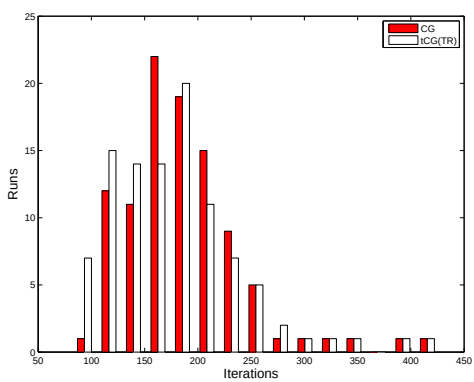
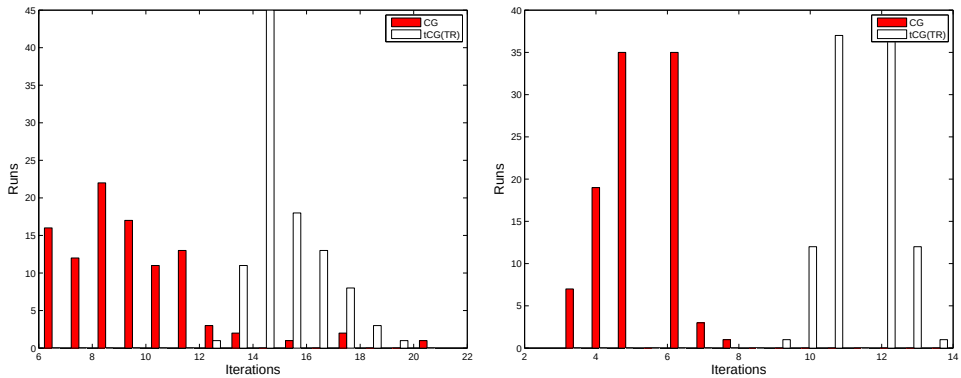


Figure 7.5: *Summary for 100 runs of the number of iterations for CG and tCG(TR) for tensors* $\mathscr{A} \in \mathbb{R}^{20\times 20\times 20}$ *with elements taken from a normal distribution with zero mean and unit standard deviation and* $R_1 = R_2 = R_3 = 5$. *The initial values were taken from the truncated HOSVD.*

n singular values can differ from run to run, leading to total number of iterations from 100 to 400. The elapsed time for CG was better than that for the trust-region algorithm.

In examples with tensors as in (7.1), the comparison between CG and the trust-region algorithm depended on the noise level and the precision of the solution. For the noise level as in (7.1) and precision equal to 10^{-6}, i.e., $\|\text{grad}\, g(\mathbf{X})\|/g(\mathbf{X}) \leq 10^{-6}$, CG converged in less iterations, see Figure 7.6. Its total computational time was also smaller. When we



(a) Uniform distribution (b) Normal distribution

Figure 7.6: *Summary for 100 runs of the number of iterations for CG and tCG(TR) for tensors* $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$ *as in (7.1) and* $R_1 = R_2 = R_3 = 5$. *The initial values were taken from the truncated HOSVD. The precision was equal to* 10^{-6}.

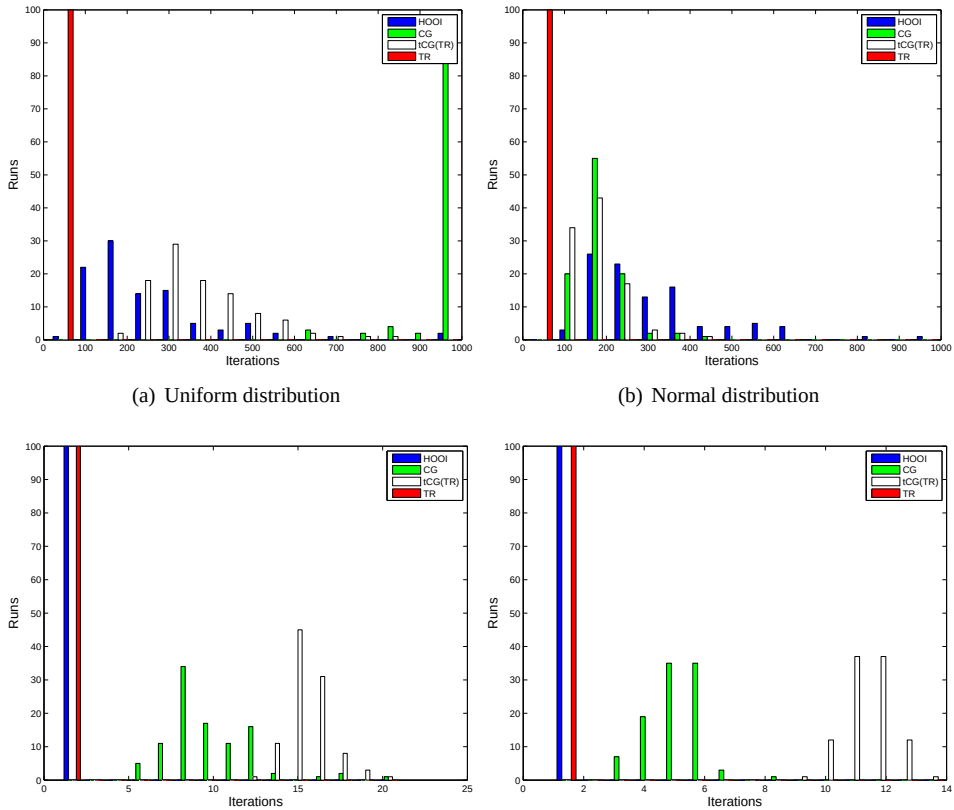
increased the precision or the noise level, the number of CG iterations increased faster than the number of tCG iterations.

The conclusion from these experiments is again that CG works better for easy problems, i.e., when the noise level is low, the precision is not too high and when the considered distribution is the normal rather than the uniform distribution.

7.4.3 Summary

In Figure 7.7, we present plots in which we compare the three algorithms (HOOI, the trust-region algorithm and CG). With respect to computational time, CG is competitive in the case of tensors with elements taken from a normal distribution with zero mean and unit standard deviation, see Figure 7.7(b). Here HOOI needs more iterations and the trust-region algorithm needs a similar number of tCG iterations. However, one CG iteration requires only a few backtracking steps for its line search whereas the computation of the Hessian of the cost function is necessary for tCG.

All three algorithms converge in general to local minima. Although started from the truncated HOSVD, different local minima were sometimes obtained by the different algorithms. If the global minimum or all local minima are desired, all available algorithms could be run with different initial points and all solutions taken into account.



(a) Uniform distribution

(b) Normal distribution

(c) Uniform distribution, $\mathscr{A}$ as in (7.1)

(d) Normal distribution, $\mathscr{A}$ as in (7.1)

Figure 7.7: *Summary of the number of iterations for 100 runs of HOOI, tCG(TR) and CG.* $\mathscr{A} \in \mathbb{R}^{20\times20\times20}$, $R_1 = R_2 = R_3 = 5$ *and the initial values were taken from the truncated HOSVD.*

7.5 Conclusions

In our study of algorithms for the low multilinear rank approximation of tensors, it was important to investigate the behavior of a CG-based algorithm. A nonlinear CG converges to stationary points under mild assumptions on the cost function. In practice, nonlinear CG converges to a local minimum unless very special initial values are started from. It is difficult to analyze the behavior of CG on a manifold theoretically but one can expect that it behaves in a similar way as nonlinear CG in $\mathbb{R}^n$. All directions are descent directions if we take the steepest descent direction whenever the proposed direction is not a descent direction. The convergence speed is not favorable but this is compensated by the fact that the iterations are extremely fast.

The performance of the CG algorithm is related to the difficulty of the problem in the following sense. If there is a clear gap between the (R_n)-th and (R_n+1)-th mode-n singular values, $n=1,2,3$, and if the first R_n singular values are of the same order, then we have a well-conditioned problem. However, in the case of very easy problems, where the original tensor is a low multilinear rank tensor slightly affected by additive noise, HOOI converges faster than CG. On the other hand, in ill-conditioned cases, the trust-region algorithm is more efficient than CG. Hence, as far as the computational time is concerned, CG seems to be competitive for examples that are not too easy and not too difficult, such as tensors with elements taken from a normal distribution with zero mean and unit standard deviation.

Even though there are algorithms in the literature that have stronger theoretical foundations than CG, it makes sense to consider CG as well. An additional observation is that the low multilinear rank approximation problem has in general many local minima. Different algorithms may converge to different minima even if they are initialized in the same way. CG may yield other minima than the other methods. The issue of local minima is further discussed in the next chapter.

Part III

Practical experience

Chapter 8

Local minima

In this chapter, we study the issue of local minima of the best low multilinear rank approximation problem (2.2). We discuss the existence of local minima and show that there can be many. The implications for real problems are also considered. The current chapter is based on [68].

The discussion on local minima is important since, in the matrix case, the low rank approximation problem has only one minimum. Moreover, for higher-order tensors, starting from the truncated HOSVD does not guarantee convergence to the global minimum. This has already been suggested in [131] and explicitly shown in [30, 37].

8.1 Existence of local minima

Let us first consider tensors $\hat{\mathcal{A}} \in \mathbb{R}^{10\times 10\times 10}$,

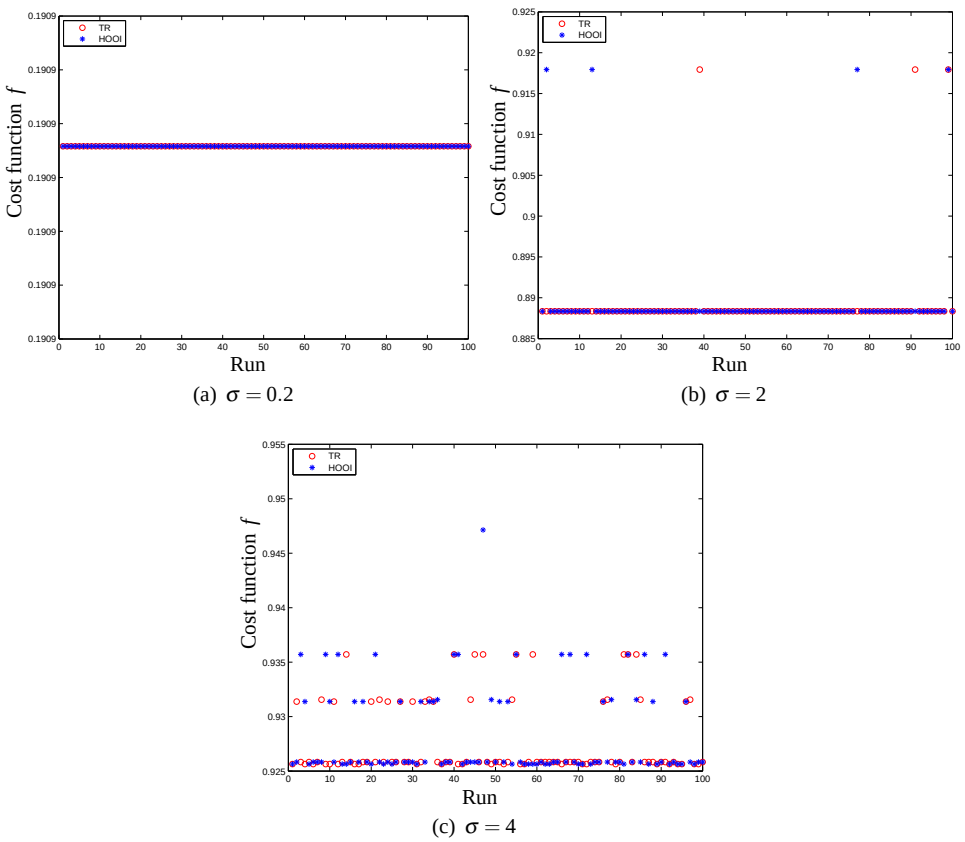
$$\hat{\mathcal{A}}(\sigma) = \mathcal{T}/\|\mathcal{T}\| + \sigma * \mathcal{E}/\|\mathcal{E}\|, \tag{8.1}$$

where $\mathcal{T} \in \mathbb{R}^{10\times 10\times 10}$ is a rank-$(2,2,2)$ tensor and $\mathcal{E} \in \mathbb{R}^{10\times 10\times 10}$ represents noise. Let $\mathcal{T}$ be obtained as a product of a random $(2,2,2)$-tensor and three random column-wise orthonormal matrices with matching dimensions, i.e., taken equal to the **Q** factors of the thin **QR** factorization of matrices with elements drawn from normal distribution with zero mean and unit standard deviation. Let the the elements of the $(2,2,2)$-tensor and of $\mathcal{E}$ also be taken from a normal distribution with zero mean and unit standard deviation. In order standardize our results as much as possible, we also normalize $\hat{\mathcal{A}}$ and work with

$$\mathcal{A} = \hat{\mathcal{A}}/\|\hat{\mathcal{A}}\|. \tag{8.2}$$

The parameter σ controls the noise level. In this setup, it makes sense to consider the rank-$(2,2,2)$ approximation of $\mathcal{A}$.

To study the convergence towards the global minimum, we examined the set of local minima computed in 100 runs, starting from different column-wise orthonormal matrices $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. In order to avoid conclusions that would only hold for one particular method, we considered two different algorithms, HOOI (Algorithm 1, Section 2.4.2) and the trust-region algorithm (Algorithm 4, Chapter 6). It can experimentally be stated that both algorithms converge globally (for all initial points) to local minima, except in very special examples that are artificially constructed. Each run was stopped if the corresponding algorithm did not converge ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations. We considered three noise levels, $\sigma = 0.2, 2, 4$. The results are presented in Figure 8.1. In the first plot, Figure 8.1(a), the noise level is low and both algorithms converged to the



(a) $\sigma = 0.2$ (b) $\sigma = 2$

(c) $\sigma = 4$

Figure 8.1: *Local minima for different noise levels. The tensors $\mathcal{A} \in \mathbb{R}^{10\times10\times10}$ are as in (8.1)–(8.2). Performed are 100 runs starting from different arbitrary column-wise orthonormal matrices $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. A run was stopped if the corresponding algorithm did not converge ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations.*

same minimum for all runs. After increasing the noise level, both algorithms found two local minima (Figure 8.1(b)). In Figure 8.1(c), there is more noise than structure and the algorithms converged to several minima.

The cost function values at the local minima also depend on the noise level σ. This dependence is illustrated in Figure 8.2. The top figure shows the distinct local minima for

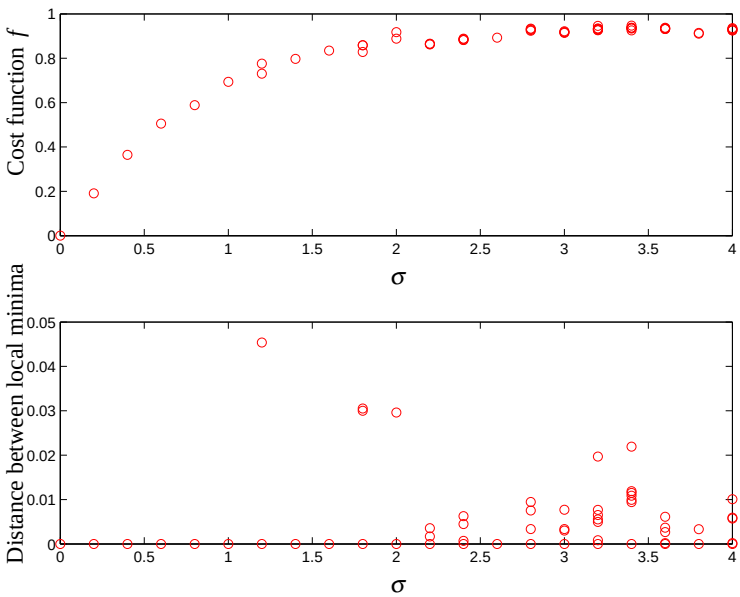


Figure 8.2: *Cost function at local minima and the distance between the local minima for different σ. The tensors $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ are as in (8.1)–(8.2). Performed are 100 runs for each σ, starting from different arbitrary column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. Each run was stopped if the corresponding algorithm did not converge ($\|grad\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations.*

each σ. The bottom figure shows the difference between the values of the cost function at the local minima. The values of largest three singular values in each mode are given in Figure 8.3 for each σ. For small σ, there is a large gap between the second and the third singular value of $\mathbf{A}_{(n)}, n = 1,2,3$. The problem is easy and few local minima appear besides the global minimum. On the other hand, for large σ, the gaps are small or nonexistent. In this case, when computing the best low multilinear rank approximation, we are looking for a structure that is not really there. The algorithms find many equally good, or equally bad, solutions. We can conclude that the number of local minima found by the algorithms depends on the difficulty of the problem. The latter is related to the distribution of the mode-n singular values of $\mathscr{A}$.

We also performed simulations with tensors $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ with elements taken from a normal distribution with zero mean and unit standard deviation. We considered

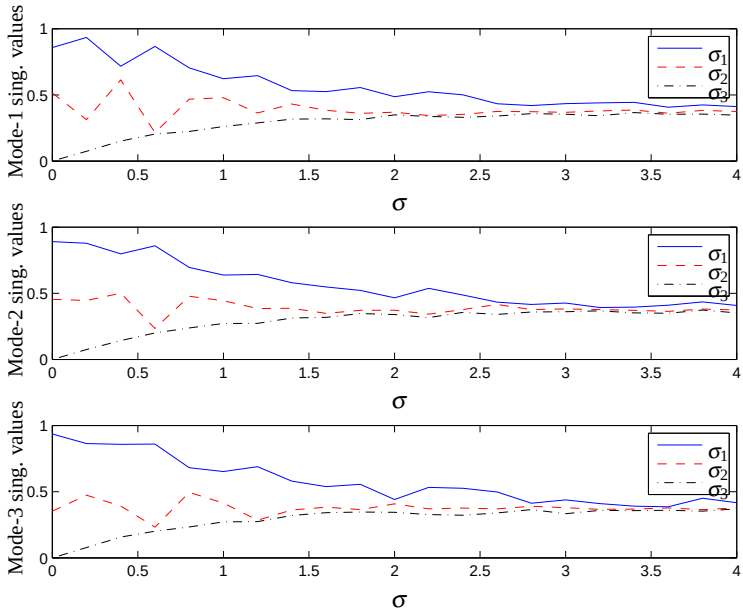
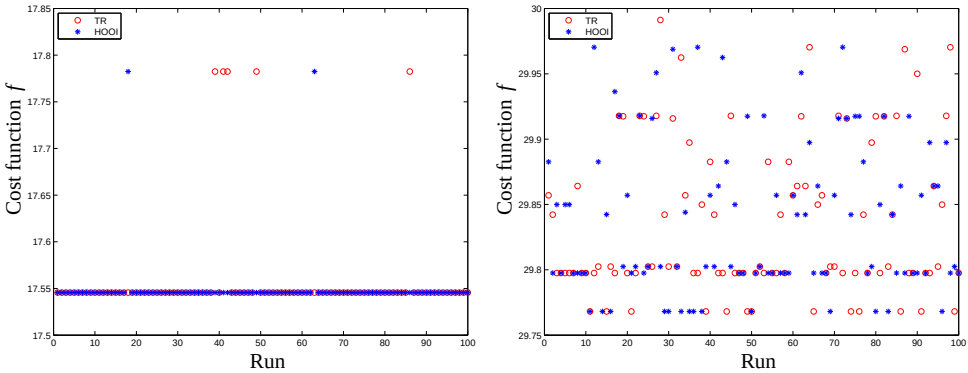


Figure 8.3: *The largest three singular values in each mode for different* σ*. The tensors* $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ *are as in (8.1)–(8.2). Performed are 100 runs for each* σ*, starting from different arbitrary column-wise orthonormal* $\mathbf{U}_0, \mathbf{V}_0$*, and* $\mathbf{W}_0$*. Each run was stopped if the corresponding algorithm did not converge (*$\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$*) in 200 iterations.*

100 runs with arbitrary initial column-wise orthonormal matrices $\mathbf{U}_0, \mathbf{V}_0, \mathbf{W}_0$ and we considered two different low multilinear rank approximations: $(R_1, R_2, R_3) = (7, 8, 9)$ and $(R_1, R_2, R_3) = (2, 2, 2)$. Both HOOI and the trust-region algorithm were run until convergence ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) or until a maximum of 200 iterations was reached. As in the previous example, both algorithms globally yield the same set of local minima. However, it was often the case that when starting from the same initial matrices, HOOI and the trust-region algorithm converged to a different local minimum. The results from one representative example with a fixed tensor are presented in Figure 8.4. The lowest "level" of points is expected to represent the global minimum. However, there is no guarantee that this is the case. Figure 8.4(a) shows the results for the rank-$(7, 8, 9)$ approximation. The number of obtained minima was smaller than in the case of rank-$(2, 2, 2)$ approximation (Figure 8.4(b)).

The fact that there are several distinct subsets of converged points in both Figure 8.1 and Figure 8.4 confirms that the differences are not caused by numerical issues but that local minima really exist. This is a very important difference with the matrix case, where the low-rank approximation problem does not have any local minima except the global minimum.



(a) $(R_1, R_2, R_3) = (7, 8, 9)$ (b) $(R_1, R_2, R_3) = (2, 2, 2)$

Figure 8.4: *Local minima for different multilinear ranks. The entries of the tensor $\mathscr{A} \in \mathbb{R}^{10\times 10\times 10}$ are taken from a normal distribution with zero mean and unit standard deviation. Performed are 100 runs starting from different arbitrary column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. Each run was stopped if the corresponding algorithm did not converge ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations.*

8.2 Properties of local minima

An interesting observation is that the actual values of the cost function at the local minima found by the algorithms in the previous section were close to each other. This can be seen in Figure 8.1 and Figure 8.4. Hence, in terms of the *cost function*, the local minima are more or less equivalent.

We also examined the subspaces generated by matrices corresponding to two different runs. We considered the subspace angles as defined in [57, §12.4.3]. Any two matrices $\mathbf{U}_1$ and $\mathbf{U}_1'$ that corresponded to two solutions with the same value of the cost function spanned the same subspace. On the other hand, the largest subspace angle between $\mathbf{U}_1$ and $\mathbf{U}_2$ corresponding to two different local minima, appeared to be close to $\pi/2$, which means that the subspaces are very different. These results also applied to $\mathbf{V}$ and $\mathbf{W}$. Hence, in terms of the *subspaces* used for the approximation, the local minima are very different.

We also examined the second largest subspace angle between the best two local optima that have been obtained in the experiment. In general, the second largest angle was much smaller than the first one but it was not always (close to) zero. Results from a simulation involving a tensor $\mathscr{A} \in \mathbb{R}^{10\times 10\times 10}$ with elements taken from a normal distribution with zero mean and unit standard deviation are presented in Table 8.1 and Figure 8.5. The table shows the results obtained by the trust-region algorithm. The imposed multilinear rank was $(2,2,2)$. Ten runs were performed, starting from different random column-wise orthonormal $\mathbf{U}_0, \mathbf{V}_0$ and $\mathbf{W}_0$. A run was stopped if the algorithm did not converge ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 100 iterations. Figure 8.5 presents the actual local minima

(a) Largest subspace angle

	1	2	3	4	5	6	7	8	9	10
1	2.01E-16	1.388476	1.182443	1.471074	1.528923	0.89849	0.89849	5.23E-10	0.89849	5.80E-10
2	1.388476	2.21E-16	1.441407	1.512369	0.809631	1.410662	1.410662	1.388476	1.410662	1.388476
3	1.182443	1.441407	2.62E-16	1.543829	1.556697	1.03913	1.03913	1.182443	1.03913	1.182443
4	1.471074	1.512369	1.543829	2.23E-16	1.356632	1.523707	1.523707	1.471074	1.523707	1.471074
5	1.528923	0.809631	1.556697	1.356632	6.31E-16	1.454416	1.454416	1.528923	1.454416	1.528923
6	0.89849	1.410662	1.03913	1.523707	1.454416	4.36E-16	2.08E-10	0.89849	1.97E-10	0.89849
7	0.89849	1.410662	1.03913	1.523707	1.454416	2.08E-10	4.70E-16	0.89849	3.99E-11	0.89849
8	5.23E-10	1.388476	1.182443	1.471074	1.528923	0.89849	0.89849	2.50E-16	0.89849	1.87E-10
9	0.89849	1.410662	1.03913	1.523707	1.454416	1.97E-10	3.99E-11	0.89849	2.08E-16	0.89849
10	5.80E-10	1.388476	1.182443	1.471074	1.528923	0.89849	0.89849	1.87E-10	0.89849	6.55E-17

(b) Second largest subspace angle

	1	2	3	4	5	6	7	8	9	10
1	0	0.431241	0.018366	1.227264	0.139027	0.069378	0.069378	0	0.069378	0
2	0.431241	0	1.146175	0.75854	0.252364	1.078618	1.078618	0.431241	1.078618	0.431241
3	0.018366	1.146175	0	0.745548	1.146822	0.080504	0.080504	0.018366	0.080504	0.018366
4	1.227264	0.75854	0.745548	0	1.06264	1.319963	1.319963	1.227264	1.319963	1.227264
5	0.139027	0.252364	1.146822	1.06264	0	0.936273	0.936273	0.139027	0.936273	0.139027
6	0.069378	1.078618	0.080504	1.319963	0.936273	0	0	0.069378	0	0.069378
7	0.069378	1.078618	0.080504	1.319963	0.936273	0	0	0.069378	0	0.069378
8	0	0.431241	0.018366	1.227264	0.139027	0.069378	0.069378	0	0.069378	0
9	0.069378	1.078618	0.080504	1.319963	0.936273	0	0	0.069378	0	0.069378
10	0	0.431241	0.018366	1.227264	0.139027	0.069378	0.069378	0	0.069378	0

Table 8.1: *Subspace angles for different matrices* $\mathbf{U}$, *obtained when the trust-region algorithm was initiated differently. The elements of* $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ *were taken from a normal distribution with zero mean and unit standard deviation. The imposed multilinear rank was* $(2,2,2)$. *Performed are 10 runs starting from different arbitrary column-wise orthonormal* $\mathbf{U}_0, \mathbf{V}_0$, *and* $\mathbf{W}_0$. *A run was stopped as soon as* $\|grad\ g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$.

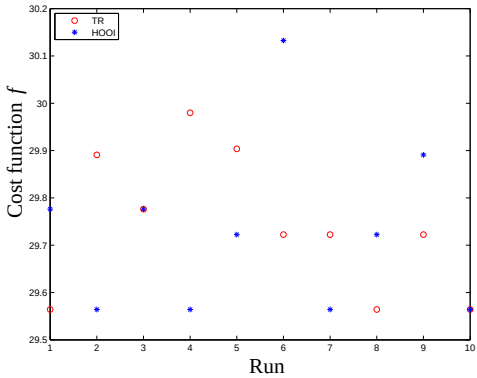


Figure 8.5: *Local minima, corresponding to Table 8.1. The tensor* $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ *had elements taken from a normal distribution with zero mean and unit standard deviation. The imposed multilinear rank was* $(2,2,2)$*. Performed were 10 runs starting from different arbitrary column-wise orthonormal* $\mathbf{U}_0, \mathbf{V}_0,$ *and* $\mathbf{W}_0$*. A run was stopped if the algorithm did not converge (*$\|grad\ g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$*) in 100 iterations. The maximum number of iterations was not reached for the trust-region algorithm in any of the runs. HOOI converged for less than 100 iterations only in run 3.*

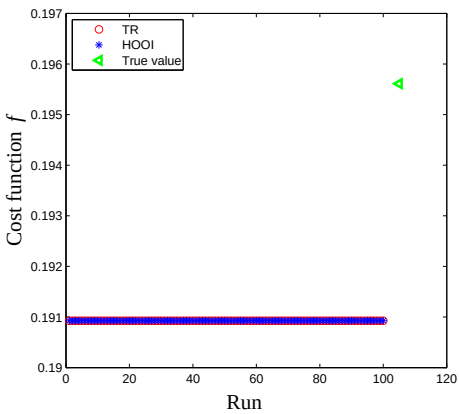
for the same example. The trust-region algorithm did not reach the maximum number of iterations in any of the runs. HOOI was stopped after 100 iterations in all runs except run 3.

The existence of local minima and the fact that the corresponding values of the cost function are similar but the corresponding subspaces are different have important consequences for applications. If the best low multilinear rank approximation is merely used as a compression tool, aiming at a reduction of the memory needed to store the tensor, taking any local minimum results in a similar compression rate as taking the global minimum itself. In this type of applications, the existence of local minima does not pose a major problem. On the other hand, in applications where the subspaces spanned by the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ are of importance, different local minima may lead to completely different results. This is for instance the case when the approximation is computed in a preprocessing step, prior to the computation of a PARAFAC decomposition.

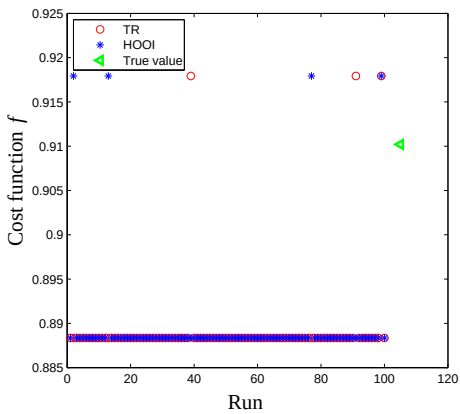
If the global minimum is of interest, one could consider different initial values and different algorithms. After obtaining enough different solutions, the best one should be retained.

An additional problem in the case where the subspaces are of interest, is that the global minimum is not necessarily the minimum that we want. Let the tensor $\mathscr{A}$ that has to be approximated be as in (8.1)–(8.2). The "true" tensor $\mathscr{T}$ is low multilinear rank. Thus, truncating the HOSVD of $\mathscr{T}$ does not lead to loss of information and the matrices from the truncated HOSVD represent the required "true" subspaces. Among the different local

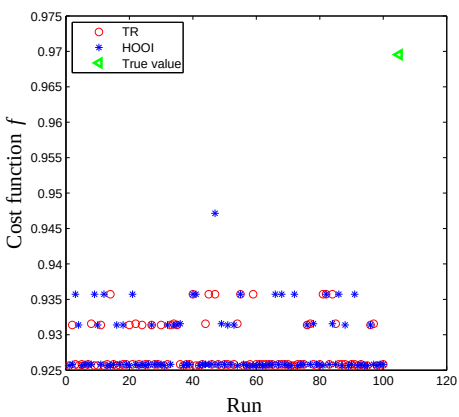
maxima of (2.3) with respect to $\mathscr{A}$, the one that yields subspaces that are closest to the "true" subspaces, is not necessarily the global maximum of (2.3), i.e., not necessarily the global minimum of (2.2). In Figure 8.6, we considered again the example corresponding to Figure 8.1. This time we also plot the value of the cost function at the point represented by the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ from the truncated HOSVD of $\mathscr{T}$, i.e., the "true" matrices.



(a) $\sigma = 0.2$



(b) $\sigma = 2$



(c) $\sigma = 4$

Figure 8.6: *Local minima for different noise levels. The tensors $\mathscr{A} \in \mathbb{R}^{10\times10\times10}$ are as in (8.1)–(8.2). Performed are 100 runs starting from different arbitrary column-wise orthonormal matrices $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$. A run was stopped if the corresponding algorithm did not converge ($\|grad\, g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) in 200 iterations. The "true" value corresponds to the subspaces spanned by the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ from the truncated HOSVD of $\mathscr{T}$.*

For $\sigma = 2$, the closest to the "true" tensor, with respect to the subspace angle [57, §12.4.3] between the "true" and obtained matrices, was the tensor corresponding to the lowest value of the cost function. For $\sigma = 4$ things were different. The local minimum with lowest value of the cost function was the closest with respect to $\mathbf{U}$ and $\mathbf{W}$. However, with respect to $\mathbf{V}$, the fourth lowest minimum was best. In our experiments, we also encountered examples where the local minima closest to $\mathbf{U}$,$\mathbf{V}$ and $\mathbf{W}$ respectively were all different. In this situation, there is actually no way to determine which of the local minima is the desired one (at least, if one does not dispose of prior knowledge about the solution). In such a case, the low multilinear rank approximation problem cannot be solved.

8.3 Using the truncated HOSVD as a starting value

Starting from the truncated HOSVD is usually a good idea. However, in general it does not guarantee convergence to the global optimum for any algorithm. Numerical examples showed that not only in specially constructed examples but also for random tensors a better local minimum can sometimes be obtained starting from another initial point. By way of illustration, recall the example corresponding to Figure 8.4. In Figure 8.7, we show also the minima obtained by HOOI and trust-region algorithm when started from the truncated HOSVD. These do not correspond to the global minimum.

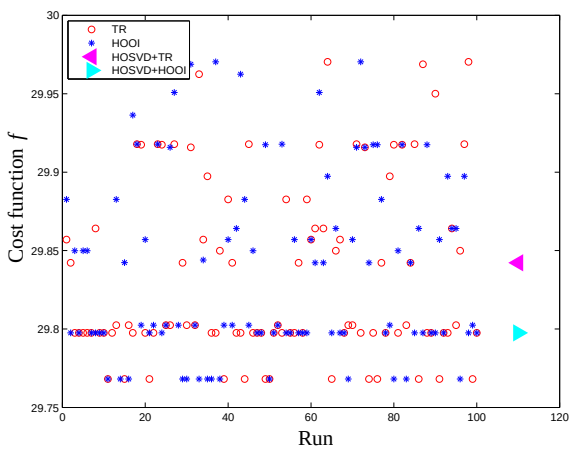


Figure 8.7: *Local minima as in Figure 8.4. Additional results for HOOI and the trust-region algorithm when started from the truncated HOSVD are presented.*

8.4 Conclusions

The purpose of this chapter was to investigate the issue of local minima of the cost function (2.2).

Different simulations show that local nonglobal minima do exist. This is a key observation since the best rank-R approximation of a matrix has a unique minimum.

In the case of tensors with low multilinear rank perturbed by a small amount of additive noise, algorithms converge to a small number of local minima. After increasing the noise level, the tensor becomes less structured and more local minima are found. This behavior is related to the distribution of the mode-n singular values. In the first case, there is a large gap between the singular values. If the gap is small or nonexistent, the best low multilinear rank approximation is a difficult problem since we are looking for a structure that is not present. In this case, there are many equally good, or equally bad, solutions.

Another interesting result concerns the values of the cost function at the different local minima. These values seem to be close to each other. Hence, in applications where the multilinear rank approximation is merely used as a compression tool for memory savings, working with a nonglobal local minimum is not too different from working with the global minimum itself.

On the other hand, the subspaces spanned by matrices $\mathbf{U}_1$ and $\mathbf{U}_2$ corresponding to two different local minima are very different and the same holds for $\mathbf{V}$ and $\mathbf{W}$. In applications where these subspaces are of importance, local minima may be an issue. The latter concerns in particular the dimensionality reduction prior to the computation of a PARAFAC decomposition. It is important to inspect the gap between the mode-n singular values in each mode in order to choose the correct values for the multilinear rank of the approximation.

An additional problem appears when the subspaces are of importance but the global minimum is not the desired solution. This could be the case when a tensor with low multilinear rank is affected by noise. The subspaces corresponding to the global minimum of (2.2) are not necessarily closest to the subspaces corresponding to the original noise-free tensor, especially if the noise level is high. This further stresses that solutions of the approximation problem (2.2) have to be interpreted with care. It may even be impossible to obtain a meaningful solution.

It is usually a good idea to start with initial values taken from the truncated HOSVD. However, this is no guarantee for convergence to the global optimum. Examples with random tensors showed that sometimes a better (i.e., with a smaller cost function value) local minimum is obtained when starting from another initial point. If the global minimum is required, different initial values and different algorithms should be considered. After the computation of several local optima, the best one should be retained.

Finally, we suggest the following procedure for dimensionality reduction of large-scale

problems. As an initial step, the smallest mode-n singular values in each mode can be inspected. The HOSVD of the original tensor can be truncated so that the mode-n singular values close to zero be discarded. In this way, the dimensions of the original tensor are reduced without losing much precision. As a second step prior to computing for example a PARAFAC decomposition, an essential dimensionality reduction via low multilinear rank approximation on an already smaller scale can be performed. The latter needs to take into account gaps between mode-n singular values.

Chapter 9

Application: Harmonic analysis

The exponentially damped sinusoidal (EDS) model [128] is a simple and widely used model in signal processing applications, including magnetic resonance spectroscopy, magnetic resonance spectroscopic imaging [137, 138, 136], electroencephalogram [28, 29] and audio processing [75]. The exponential data fitting problem consists of estimating the parameters of the EDS model, given a number of samples. Matrix-based methods for the estimation of the unknown parameters are well-known. In [111, 112, 110, 58], tensor-based algorithms for the single-channel, multi-channel, and decimative case were proposed. These algorithms make use of the best low multilinear rank approximation of tensors.

In this chapter, we consider in particular the decomposition of a signal in exponential components in the multi-channel case. Given is a set of signals with identical poles but different complex amplitudes. The goal is to estimate the poles of the model, i.e., the frequencies and the damping factors. We define the problem formally and briefly recall the tensor algorithm from [111]. We compare the performance of the tensor-based and the corresponding matrix-based method [139] for the case of closely spaced poles. We look for factors that affect the performance of the tensor method, in order to see in which cases it is better to use the latter instead of the matrix-based method. Our findings are also summarized in [73].

9.1 Introduction

We consider a decomposition of a signal in a sum of K complex damped or undamped sinusoids. K is called the *model order*. Given N samples x_n, $n = 0, \ldots, N-1$,

$$x_n = \sum_{k=1}^{K} a_k \exp\{j\varphi_k\} \exp\{(-\alpha_k + 2j\pi\nu_k)\, t_n\} = \sum_{k=1}^{K} c_k z_k^n,$$

where $j = \sqrt{-1}$, $t_n = n\Delta t$ and Δt is the sampling interval, the goal is to accurately estimate the parameters, i.e., the *amplitudes* a_k, the *phases* φ_k, the *damping factors* α_k, and the *frequencies* ν_k, or, equivalently, the *complex amplitudes* $c_k = a_k \exp\{j\varphi_k\}$ and the *poles* $z_k = \exp\{(-\alpha_k + 2j\pi\nu_k)\, \Delta t\}$.

Matrix-based methods for the estimation of the unknown parameters are well-known [50]. There exist maximum likelihood methods, linear prediction methods, methods based on higher-order statistics and subspace based methods.

We consider multi-channel signals and focus on subspace methods, in particular the HTLSstack algorithm (Hankel total least squares method for multi-channel signal) [139], which is a variant of HSVD (Hankel SVD) [17] and HTLS [136]. In [111, 110], a related tensor-based algorithm is proposed. Another tensor-based algorithm was recently presented in [58]. Of interest for this thesis is the case of closely spaced poles. It has been shown by simulations [112] that in the decimative case the tensor-based method is more robust against noise than its matrix-based counterpart. Our goal is to compare the matrix [139] and the tensor-based [111, 110] method in the multi-channel case. We assume there are Q channels. The signals in the different channels have identical poles but different complex amplitudes, i.e., for $n = 0, \ldots, N-1$, $q = 1, \ldots, Q$,

$$x_n^{(q)} = \sum_{k=1}^{K} a_k^{(q)} \exp\{j\varphi_k^{(q)}\} \exp\{(-\alpha_k + 2j\pi\nu_k)\, t_n\} + e_n^{(q)} = \sum_{k=1}^{K} c_k^{(q)} z_k^n + e_n^{(q)}, \quad (9.1)$$

where $e_n^{(q)}$ represents noise.

9.2 Theoretical background

In this chapter, we will often use Hankel and Vandermonde matrices. An $(L \times M)$-*Hankel matrix* with elements x_i, $i = 0, \ldots, N-1$, has the following form

$$\begin{pmatrix} x_0 & x_1 & x_2 & \cdots & x_{M-1} \\ x_1 & x_2 & \iddots & \cdots & \vdots \\ x_2 & \iddots & \iddots & \cdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & x_{N-2} \\ x_{L-1} & \cdots & \cdots & x_{N-2} & x_{N-1} \end{pmatrix}.$$

All elements along any antidiagonal are the same. An $(m \times n)$-*Vandermonde matrix* is a matrix of the form

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_1^{m-1} & x_2^{m-1} & \cdots & x_n^{m-1} \end{pmatrix}.$$

Every column of such a matrix is a geometric progression.

In the tensor-based algorithms for parameter estimation, the low multilinear rank approximation of tensors plays an essential role. One can choose values for the mode-n ranks of the approximation such that the performance of the tensor method is optimal. In the multi-channel case, the main steps of the tensor HO-HTLSstack algorithm are presented in Algorithm 7 [111]. HO-HTLS stands for "Higher-order Hankel total least squares".

Some remarks about Algorithm 7 are in order[1]:

- If the model order K is not known a priori, it can be estimated as the number of significant singular values of the matrices $\mathbf{H}_q$.
- In the case of real signals, K should be substituted by $2K$. This is because a real sinusoid can be represented as the sum or difference of 2 complex conjugated exponentials.
- In step 4, $\overline{\mathbf{Z}}$ is estimated from $\mathbf{U}^{\uparrow} \approx \mathbf{U}_{\downarrow}\,\overline{\mathbf{Z}}$ in the TLS (total least squares) sense. The matrix $\mathbf{V}$ could also be used instead of $\mathbf{U}$.

[1] $\hat{z}_k, \hat{c}_k^{(q)}, \hat{\alpha}_k, \hat{\nu}_k, \hat{a}_k^{(q)}, \hat{\varphi}_k^{(q)}$ are the estimates of the true values $z_k, c_k^{(q)}, \alpha_k, \nu_k, a_k^{(q)}, \varphi_k^{(q)}$, $k = 1, 2, \ldots, K$, $q = 1, 2, \ldots, Q$.

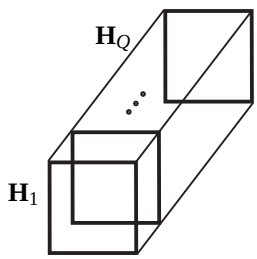
Algorithm 7 HO-HTLSstack

1: Construct $(L \times M)$-Hankel matrices $\mathbf{H}_q$ from the samples $x_n^{(q)}, n = 0, \ldots, N-1, q = 1, \ldots, Q$,

$$\mathbf{H}_q = \begin{pmatrix} x_0^{(q)} & x_1^{(q)} & x_2^{(q)} & \cdots & x_{M-1}^{(q)} \\ x_1^{(q)} & x_2^{(q)} & \iddots & \cdots & \vdots \\ x_2^{(q)} & \iddots & \iddots & \cdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & x_{N-2}^{(q)} \\ x_{L-1}^{(q)} & \cdots & \cdots & x_{N-2}^{(q)} & x_{N-1}^{(q)} \end{pmatrix},$$

i.e., the samples from each channel $q, q = 1, \ldots, Q$, are arranged in an $(L \times M)$-Hankel matrix with $N = L + M - 1, L > K$. These matrices should be as square as possible.

2: Stack the matrices $\mathbf{H}_q$ in an $(L \times M \times Q)$-tensor $\mathscr{H}$ as illustrated.



3: Let $R_3 \leq \min(K, Q)$. Compute the best rank-(K, K, R_3) approximation $\hat{\mathscr{H}}$ of $\mathscr{H}$,

$$\hat{\mathscr{H}} = \mathscr{S} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}.$$

4: Compute $\overline{\mathbf{Z}}$ from

$$\mathbf{U}^{\uparrow} \approx \mathbf{U}_{\downarrow} \overline{\mathbf{Z}},$$

where $\mathbf{U}^{\uparrow}$ (resp. $\mathbf{U}_{\downarrow}$) is derived from $\mathbf{U}$ by deleting it's first (resp. last) row.

5: Compute the eigenvalues λ_k of $\overline{\mathbf{Z}}$. They yield an estimate of the signal poles.

6: Estimate the damping factors $\hat{\alpha}_k$ and frequencies $\hat{\nu}_k$ from

$$\lambda_k = \hat{z}_k = \exp\{(-\hat{\alpha}_k + 2\pi j \hat{\nu}_k)\, \Delta t\}.$$

7: Estimate the complex amplitudes $\hat{c}_k^{(q)} = \hat{a}_k^{(q)} \exp\{j \hat{\varphi}_k^{(q)}\}$ as the least squares solution of

$$\begin{pmatrix} 1 & \cdots & 1 \\ \hat{z}_1^1 & \cdots & \hat{z}_K^1 \\ \hat{z}_1^2 & \cdots & \hat{z}_K^2 \\ \vdots & \vdots & \vdots \\ \hat{z}_1^{N-1} & \cdots & \hat{z}_K^{N-1} \end{pmatrix} \begin{pmatrix} c_1^{(q)} \\ c_2^{(q)} \\ \vdots \\ c_K^{(q)} \end{pmatrix} = \begin{pmatrix} x_0^{(q)} \\ x_1^{(q)} \\ x_2^{(q)} \\ \vdots \\ x_{N-1}^{(q)} \end{pmatrix}.$$

- The relation between $\overline{\mathbf{Z}}$ and the diagonal matrix $\mathbf{Z} = \mathrm{diag}(z_1, z_2, \ldots, z_K)$ with the poles $z_i, i = 1, \ldots, K$ on its diagonal, is as follows [111, 110]

  $$\overline{\mathbf{Z}} = \mathbf{Q}^{-1}\mathbf{Z}\mathbf{Q},$$

  where $\mathbf{Q} \in \mathbb{C}^{K\times K}$ is nonsingular. The eigenvalues of $\overline{\mathbf{Z}}$ yield an estimate of the signal poles.

- In step 3, the low multilinear rank approximation is computed by HOOI. Alternatively, our geometric Newton, trust-region and conjugate gradient based algorithms could also be used.

We mention that in the absence of noise, it is possible to decompose the tensor in step 2 of Algorithm 7 as

$$\mathscr{H} = \mathscr{I} \bullet_1 \mathbf{Z}_1 \bullet_2 \mathbf{Z}_2 \bullet_3 \mathbf{C}, \tag{9.2}$$

where $\mathscr{I}$ is a diagonal $(K \times K \times K)$-tensor with ones on the diagonal, $\mathbf{Z}_1, \mathbf{Z}_2$ are Vandermonde matrices generated by the poles $z_1, \ldots, z_k$,

$$\mathbf{Z}_1 = \begin{pmatrix} 1 & \cdots & 1 \\ z_1 & \cdots & z_K \\ z_1^2 & \cdots & z_K^2 \\ \vdots & \vdots & \vdots \\ z_1^{L-1} & \cdots & z_K^{L-1} \end{pmatrix}, \quad \mathbf{Z}_2 = \begin{pmatrix} 1 & \cdots & 1 \\ z_1 & \cdots & z_K \\ z_1^2 & \cdots & z_K^2 \\ \vdots & \vdots & \vdots \\ z_1^{M-1} & \cdots & z_K^{M-1} \end{pmatrix}, \tag{9.3}$$

and $\mathbf{C}$ is the matrix that contains the complex amplitudes

$$\mathbf{C} = \begin{pmatrix} c_1^{(1)} & \cdots & c_K^{(1)} \\ c_1^{(2)} & \cdots & c_K^{(2)} \\ c_1^{(3)} & \cdots & c_K^{(3)} \\ \vdots & \vdots & \vdots \\ c_1^{(Q)} & \cdots & c_K^{(Q)} \end{pmatrix}.$$

The HOSVD of the tensor $\mathscr{H}$ leads to a decomposition similar to (9.2) in the sense that the column-wise orthonormal matrices of the truncated HOSVD of $\mathscr{H}$ span the same subspaces as the matrices $\mathbf{Z}_1, \mathbf{Z}_2$ and $\mathbf{C}$. In the presence of noise, the best rank-(K, K, R_3), $R_3 \leq \min(K, Q)$ approximation can be used instead of HOSVD for obtaining a similar decomposition.

The matrix algorithm that we consider is HTLSstack [139]. It follows the same steps as Algorithm 7 except for step 2 and step 3. In step 2, the matrices $\mathbf{H}_q$ are put one after the other in a wide $(L \times MQ)$-matrix $\mathbf{H}$. In Step 3, the truncated SVD of $\mathbf{H}$ is computed. The

matrix $\mathbf{U}$ in step 4 is taken to be the K-dimensional left dominant subspace of $\mathbf{H}$. The decomposition of $\mathbf{H}$ in the sense of (9.2) is

$$\mathbf{H} = \mathbf{Z}_1[\mathbf{C}_1\mathbf{Z}_2^T \;\; \mathbf{C}_2\mathbf{Z}_2^T \;\cdots\; \mathbf{C}_Q\mathbf{Z}_2^T],$$

where

$$\mathbf{H}_q = \mathbf{Z}_1\mathbf{C}_q\mathbf{Z}_2^T = \mathbf{Z}_1\mathrm{diag}(c_1^{(q)}, c_2^{(q)}, \ldots, c_K^{(q)})\mathbf{Z}_2^T$$

are the Vandermonde decompositions of $\mathbf{H}_q$, $q = 1, 2, \ldots, Q$.

If $K < Q$, the tensor $\mathscr{H}$ is mode-3 rank deficient. This property is also exploited by the matrix algorithm. However, in the tensor algorithm, the structure of the data can be better exploited. In step 3 of Algorithm 7, we could reduce the mode-3 rank even more in case of ill-conditioning. Thus, better performance of the tensor algorithm is expected for certain types of examples.

9.3 Simulations

We considered examples as in [98, 129] with two peaks ($K = 2$) and with closely spaced poles. We first considered frequencies $\nu_1 = 0.2, \nu_2 = 0.205$, and no damping ($\alpha_1 = \alpha_2 = 0$). The phase was set equal to zero, i.e., $\varphi_k^{(q)} = 0$ and the amplitudes $a_k^{(q)}$ were such that the matrix $\mathbf{C}$ was ill-conditioned with condition number 100. The other parameters were set as follows: $t_n = n, k = 1, 2$, $q = 1, \ldots, 13$, $n = 0, \ldots, 24$. Complex circular (with equal variance for the real and imaginary part) white Gaussian noise was added. The experiments were carried out in MATLAB.

Results of Monte-Carlo simulations of 100 independent runs are shown in Figure 9.1 for noise standard deviations between 0 and 1. RRMSE is the relative root mean squared error of the frequency estimation

$$\mathrm{RRMSE} = \left(\frac{1}{N_{\mathrm{runs}}} \sum_{i=1}^{N_{\mathrm{runs}}} |\nu - \hat{\nu}_i)|^2 \right)^{1/2} \frac{100}{\nu} \quad [\%],$$

where $\hat{\nu}_i$ is the estimate of ν in the ith run and N_{runs} is the number of runs. The mode-3 singular values of $\mathscr{H}$ indicate that the matrix $\mathbf{C}$ is ill-conditioned. In step 3 of Algorithm 7 we can exploit the ill-conditioning of $\mathbf{C}$ and set $R_3 = 1$. It can be seen that the tensor method with reduced mode-3 rank outperformed the matrix method. Extensive simulations showed that if the matrix of the complex amplitudes $\mathbf{C}$ is ill-conditioned, the improvement is considerable. If $\mathbf{C}$ is well-conditioned, then the matrix algorithm has better performance than the tensor method with reduced rank. The tensor method with equal mode-n ranks had similar behavior as the matrix method HTLSstack [139].

Next, we introduced a mild damping factor in one of the peaks. We performed similar simulations as above but with $\alpha_1 = 0$, $\alpha_2 = 0.4$. The results are presented in Figure 9.2.

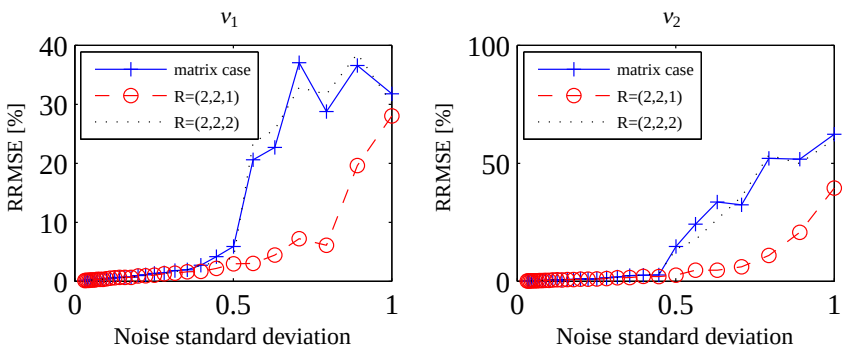


Figure 9.1: *The effect of reducing the mode-*3 *rank in step 3 of Algorithm 7, in the case of ill-conditioned* **C***, no damping.*

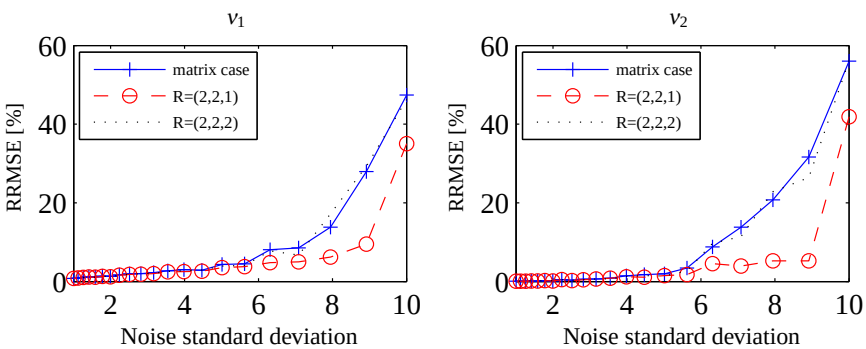
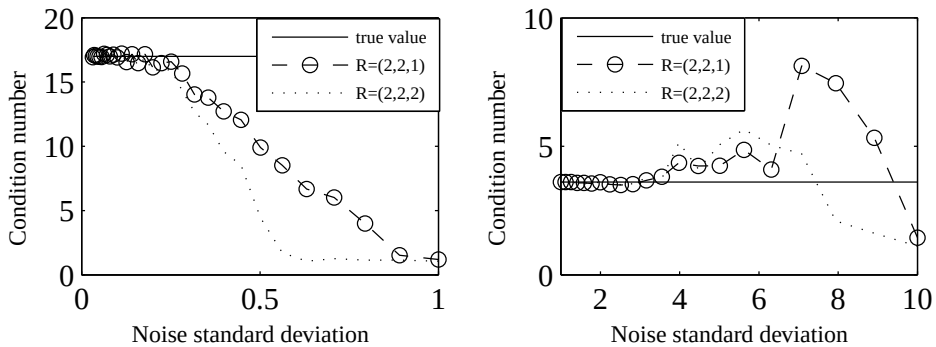


Figure 9.2: *The effect of reducing the mode-*3 *rank in step 3 of Algorithm 7, in the case of ill-conditioned* **C** *after introducing damping factor.*

The damping improved the performance of the methods. The noise level up to which the algorithms worked well, was considerably higher than in the first experiment. The best method was still the tensor method with reduced mode-3 rank. The better performance in the second experiment is due to the better condition number of the Vandermonde matrices generated by the poles (9.3). In the first example, the poles were close to each other. Introducing a mild damping factor in one of the peaks increases the difference between the two poles. Figure 9.3 shows that the condition number of the Vandermonde matrices decreases due to the introduction of the damping factor. The horizontal line shows the true value of the condition number. The other lines show the condition number of the estimated matrix.



(a) No damping

(b) Damping factors 0 and 0.4

Figure 9.3: *Condition number of the Vandermonde matrix generated by the poles in case of no damping and in case of small damping factor for one of the peaks.*

9.4 Conclusions

We considered the EDS model in the multi-channel case and under the condition that the poles of the model are closely spaced. This problem is more difficult than the case where the poles are well separated. The goal of this study was a comparison of the performance of a matrix-based and a tensor-based method. None of them turned out to always outperform the other one. The main question was then which factors affect the performance of the tensor method, i.e., in which cases is it better to use the tensor-based method than the matrix-based method.

In the tensor-based algorithm, the low multilinear rank approximation of tensors plays an essential role. One can choose the mode-3 rank in such a way that the performance is optimal. We used this technique when the matrix containing the amplitudes was ill-conditioned. Our numerical experiments indicated that taking a lower value for the mode-3 rank than for the mode-1 and mode-2 ranks improves the performance of the tensor method to the extent that it outperforms the matrix method.

A factor that affects the performance of both methods is the damping. A mild damping factor in one of the peaks improves the condition number of the Vandermonde matrix generated by the poles and thus improves the overall performance.

The tensor algorithm HO-HTLSstack is based on computing a low multilinear rank approximation of a third-order tensor. The original algorithm and our simulations used HOOI for this purpose. Alternatively, the algorithms proposed in this thesis – the geometric Newton, trust-region and conjugate gradient based algorithms – could also be used.

Part IV

Closing

Chapter 10

Conclusions and open problems

10.1 General conclusions of the thesis

This thesis is a combination of two hot topics. The main problem of the thesis, the best low multilinear rank approximation of higher-order tensors, is a key problem in *multilinear algebra* and has various applications. We proposed solutions based on *optimization on manifolds*, because of symmetry properties of the cost functions (2.3) and (5.4). The fact that the cost functions are invariant under right multiplication of the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$ by orthogonal matrices prohibits potential algorithms from converging to a particular solution. Working on quotient manifolds, we managed to isolate the solutions and made the work of "standard" optimization algorithms easier.

We developed four algorithms, one for a matrix problem and three for the main tensor approximation problem in this thesis mentioned above. We studied their numerical properties and compared them with each other and with existing methods, in particular with HOOI (Algorithm 1). The optimization methods on which we based our methods were Newton's method, trust-region and conjugate gradients. The matrix algorithm proposed in Chapter 4 is an intermediate step towards Newton's method Algorithm 3 for the tensor approximation problem. However, it is on its own an interesting algorithm for Oja's vector field.

It is difficult to say which algorithm is the best. All algorithms have their advantages and disadvantages. Depending on the application, the dimensions of the tensor, the required precision and the restrictions of time, one of the algorithms can be the method of choice. Our Newton algorithm has local quadratic convergence rate but might diverge or

converge to a saddle point or a maximum instead of a minimum. Moreover, it needs a good starting point. A well-chosen stopping criterion for the inner iteration of the trust-region algorithm can lead to an algorithm with local quadratic convergence. The computational cost per iteration is competitive with the one of HOOI (Algorithm 1), which has only linear local convergence. Moreover, convergence of the trust-region algorithm to a minimum is (almost always) guaranteed. On the other hand, the conjugate gradient based algorithm has much cheaper iterations but lacks solid theoretical proofs. The advantages and disadvantages of each of the three algorithms are provided at the end of the corresponding chapters. We summarize the most important features of the algorithms in Table 10.1.

It can make sense to apply several algorithms to the same problem. For example, if one wishes to inspect several local minima, one strategy is to run all available algorithms, starting from enough initial points and in this way (hopefully) obtain a full set of solutions. Due to the different character of the algorithms, they often find different solutions even when starting from the same initial values. In any case, they can also be considered as a theoretical achievement.

Finally, we mention some additional results that we have obtained. We investigated the issue of local minima of the low multilinear rank approximation problem. This study concerns the problem itself and does not depend on the actual algorithm. There are important consequences for whole classes of applications. Our conclusion here is that the users should be very careful when deciding whether or not the usage of such an approximation is meaningful. The higher-order singular values may provide relevant information in this respect.

We have also investigated the issue of how to choose the multilinear rank in an applied parameter estimation problem. We compared the performance of matrix and tensor based algorithms for multi-channel harmonic analysis and found factors that affect the performance of the tensor method and make it perform better than the simpler matrix algorithm in ill-conditioned cases.

10.2 Future work and open problems

Improvements of the proposed algorithms

Some of the algorithms proposed in this thesis can be further improved. For example, in Step 4 of Algorithm 6, exact line search could be considered instead of the Armijo point. Such an improvement could lead to a better convergence behavior of the algorithm. Also in Algorithm 4, once a search direction is fixed, an exact line search could be performed. Such a modification could be useful especially for the first outer steps of the algorithm. A modification of the trust-region based algorithm in the sense of implicit trust-region method [15] could also be beneficial. It might as well be interesting to compare

	HOOI (Algorithm 1)	Newton (Algorithm 3)	TR (Algorithm 4)	CG (Algorithm 6)	Newton [51]	quasi-Newton [118]
global/local convergence	(global)	local	global	(global)	local	(global)
convergence to	min, (saddle point), ((max))	stationary point	min, (saddle point), ((max))	min, (saddle point), ((max))	stationary point	min, (saddle point), ((max))
local convergence speed	linear	quadratic	superlinear up to quadratic	$\begin{pmatrix} d\text{-step} \\ \text{quadratic} \end{pmatrix}$	quadratic	superlinear
cost/iteration	$O(I^3R+IR^4+R^6)$	$O(I^3R^3)$	$\leq O(I^3R^3)$	$(\sim O(I^3R))$	$O(I^3R^3)$	$O(I^3R^2)$ $O(I^3R+I^2R^3)$ $O(I^3R)$
Manifold	–	$\mathbb{R}_*^{n\times p}/O_p$	Grassmann	Grassmann	Grassmann	Grassmann
R	–	(3.11)	(3.10)	(3.10)	geodesics	geodesics
$\mathscr{T}$	–	–	–	(3.12)	–	parallel transp.
monotonically decreasing?	yes	no	yes	yes	no	depends on parameters

Table 10.1: *Summary of the main features of HOOI (Algorithm 1), Newton's algorithm (Algorithm 3), trust-region algorithm (Algorithm 4), conjugate gradient algorithm (Algorithm 6), Newton's algorithm [51] and quasi-Newton algorithm [118].*

our algorithms with the general optimization routines from the MATLAB optimization toolbox.

Local minima

The issue of local minima is crucial for certain types of applications. The consequences of their existence have to be further analyzed. It would be interesting to determine exactly how many local minima there are. One should also examine in more detail when the best low multilinear rank approximation of a higher-order tensor gives a meaningful solution. The higher-order singular values yield relevant information but the characterization of the condition of the problem in terms of the gap between the higher-order singular values may not be the most accurate. As an alternative, one might consider the difference between the norm of the approximation and the norm obtained when one of the mode-n ranks is increased by one.

Structured tensors

So far, we have considered tensors without structure or tensors with low multilinear rank affected by additive noise. In applications, there are often restrictions in the sense of

- symmetry,
- sparsity,
- non-negative entries,
- Hankel-type structure,
- Toeplitz-type structure,
- etc.

The algorithms proposed in this thesis can be applied to such tensors. However, it might be useful to exploit the structure at hand. Fully and partially symmetric versions of our algorithms are quite easily obtained.

Real applications

It would be interesting to consider real applications. So far, we have tested our algorithms only on simulated data. Each application has its properties and adjustments of the algorithms could be necessary.

For example, in our research group, we have real EEG (electroencephalogram) data at our disposal. EEG measures electrical brain activity over time. Analysis of EEG data gives insight in neurological disorders. Algorithms based on PARAFAC exist for analyzing EEG [43, 44, 8]. In its turn, PARAFAC makes use of the best low multilinear rank approximation of tensors. Currently, HOOI or truncated HOSVD are used for this purpose. Our algorithms might be a worthwhile substitute for HOOI. Moreover, the HOOI or truncated HOSVD are used without any check of the obtained subspaces. As we have seen in this thesis, this might lead to false interpretations because of the different nature of the local minima. Better treatment of this issue is required.

Other low-rank approximations

We have studied the best low multilinear rank approximation of a higher-order tensor. As mentioned in Section 2.1.2, there are other possible generalizations of matrix rank. More specifically, there is a need for more efficient and robust algorithms for computing the best low rank-R approximation of higher-order tensors.

In particular, as a fundamental problem, the best rank-1 approximation of higher-order tensors [37, 146, 80] is useful in many applications. It is interesting that this approximation can be considered as a special case of both main approximations, namely the rank-R and the rank-(R_1, R_2, R_3) approximation, when $R = R_1 = R_2 = R_3 = 1$.

As mentioned in Section 2.4.1, there are generalizations of the matrix SVD other than HOSVD. These generalizations focus on properties such as maximum diagonality of the core tensor [24, 38, 96] or imposing orthogonal [81] or symmetric [26] constraints on the PARAFAC decomposition. Of particular importance are the recent block term decompositions [32, 33, 40], which unify the Tucker/HOSVD and CANDECOMP/PARAFAC structures. The currently used algorithms for these decompositions are mainly of alternating least squares type.

Conceptually faster algorithms have to be developed for all these types of low-rank approximations. The PARAFAC model is unique under mild conditions [85, 122, 126]. However, the uniqueness is with respect to the rank-1 terms and not with respect to the vectors, which outer products yield the terms. There are indeterminacies in the sense of scaling and permutation. It would be interesting to exploit this property and investigate if working on manifolds could be beneficial. However, the mentioned indeterminacies are weaker than the invariance property (3.1) of the best low multilinear rank approximation problem. In any case, working on a manifold or not, Newton, trust-region and conjugate gradient methods for solving the PARAFAC problem could be investigated.

Bibliography

[1] P.-A. Absil, C. G. Baker, and K. A. Gallivan. Trust-region methods on Riemannian manifolds. *Found. Comput. Math.*, 7(3):303–330, July 2007.

[2] P.-A. Absil, A. Edelman, and P. Koev. On the largest principal angle between random subspaces. *Linear Algebra Appl.*, 414(1):288–294, 2006.

[3] P.-A. Absil, M. Ishteva, L. De Lathauwer, and S. Van Huffel. A geometric Newton method for Oja's vector field. *Neural Comput.*, 21(5):1415–1433, May 2009.

[4] P.-A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Appl. Math.*, 80(2):199–220, January 2004.

[5] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, January 2008.

[6] P.-A. Absil, R. Sepulchre, and R. Mahony. Continuous-time subspace flows related to the symmetric eigenproblem. *Pac. J. Optim.*, 4(2):179–194, 2008.

[7] P.-A. Absil, R. Sepulchre, P. Van Dooren, and R. Mahony. Cubically convergent iterations for invariant subspace computation. *SIAM J. Matrix Anal. Appl.*, 26(1):70–96, 2004.

[8] E. Acar, C. A. Bingol, H. Bingol, R. Bro, and B. Yener. Multiway analysis of epilepsy tensors. *ISMB 2007 Conference Proceedings, Bioinformatics*, 23(13):i10–i18, 2007.

[9] R. L. Adler, J.-P. Dedieu, J. Y. Margulies, M. Martens, and M. Shub. Newton's method on Riemannian manifolds and a geometric model for the human spine. *IMA J. Numer. Anal.*, 22(3):359–390, July 2002.

[10] M. Al-Baali. Descent property and global convergence of the Fletcher-Reeves method with inexact line search. *IMA J. Numer. Anal.*, 5:121–124, 1985.

[11] F. Alvarez, J. Bolte, and J. Munier. A unifying local convergence result for Newton's method in Riemannian manifolds. *Foundations of Computational Mathematics*, 8(2):197–226, 2008.

[12] C. A. Andersson and R. Bro. Improving the speed of multi-way algorithms: Part I. Tucker3. *Chemometrics and Intelligent Laboratory Systems*, 42(1):93–103, 1998.

[13] C. A. Andersson and R. Bro. The N-way toolbox for MATLAB. *Chemometrics and Intelligent Laboratory Systems*, 52(1):1–4, 2000. See also http://www.models.kvl.dk/source/nwaytoolbox/.

[14] R. Badeau and R. Boyer. Fast multilinear singular value decomposition for structured tensors. *SIAM J. Matrix Anal. Appl.*, 30(3):1008–1021, 2008. Special Issue on Tensor Decompositions and Applications.

[15] C. G. Baker, P.-A. Absil, and K. A. Gallivan. An implicit trust-region method on Riemannian manifolds. *IMA J. Numer. Anal.*, 28(4):665–689, 2008.

[16] P. Baranyi, A. R. Varkonyi-Koczy, Y. Yam, and R. J. Patton. Adaptation of TS fuzzy models without complexity expansion: HOSVD-based approach. *IEEE Trans. Instrumentation and Measurement*, 54(1):52–60, 2005.

[17] H. Barkhuijsen, R. de Beer, and D. van Ormondt. Improved algorithm for noniterative time-domain model fitting to exponentially damped magnetic resonance signals. *J. Magn. Reson.*, 73(3):553–57, 1987.

[18] G. Beylkin and M. J. Mohlenkamp. Numerical operator calculus in higher dimensions. *Proc. Natl. Acad. Sci. USA*, 99:10246–10251, 2002.

[19] G. Beylkin and M. J. Mohlenkamp. Algorithms for numerical analysis in high dimensions. *SIAM J. Sci. Comput.*, 26:2133–2159, 2005.

[20] C. Caiafa and A. Cichocki. Reconstructing matrices and tensors from few rows and columns. In *Proceedings of 2009 International Symposium on Nonlinear Theory and its Applications*, 2009. In press.

[21] J. Carroll and J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[22] J. Chen and Y. Saad. On the tensor SVD and the optimal low rank orthogonal approximation of tensors. *SIAM J. Matrix Anal. Appl.*, 30(4):1709–1734, 2009.

[23] T. P. Chen and S. Amari. Unified stabilization approach to principal and minor components extraction algorithms. *Neural Networks*, 14(10):1377–1387, 2001.

[24] P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36(3):287–314, 1994. Special issue on Higher-Order Statistics.

[25] P. Comon. Tensor decompositions. In J. G. McWhirter and I. K. Proudler, editors, *Mathematics in Signal Processing V*, pages 1–24. Clarendon Press, Oxford, UK, 2002.

[26] P. Comon, G. Golub, L.-H. Lim, and B. Mourrain. Symmetric tensors and symmetric tensor rank. *SIAM J. Matrix Anal. Appl.*, 30(3):1254–1279, 2008. Special Issue on Tensor Decompositions and Applications.

[27] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, PA, 2000.

[28] W. De Clercq. *Advanced preprocessing techniques and nonlinear signal analysis applied to scalp electroencephalograms for the prediction of epileptic seizures*. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, May 2005.

[29] W. De Clercq, B. Vanrumste, J.-M. Papy, W. Van Paesschen, and S. Van Huffel. Modeling common dynamics in multichannel signals with applications to artifact and background removal in EEG recordings. *IEEE Trans Biomed Eng.*, 52(12):2006–2015, 2005.

[30] L. De Lathauwer. *Signal Processing Based on Multilinear Algebra*. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, September 1997.

[31] L. De Lathauwer. A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization. *SIAM J. Matrix Anal. Appl.*, 28(3):642–666, December 2006.

[32] L. De Lathauwer. Decompositions of a higher-order tensor in block terms — Part I: Lemmas for partitioned matrices. *SIAM J. Matrix Anal. Appl.*, 30(3):1022–1032, 2008. Special Issue on Tensor Decompositions and Applications.

[33] L. De Lathauwer. Decompositions of a higher-order tensor in block terms — Part II: Definitions and uniqueness. *SIAM J. Matrix Anal. Appl.*, 30(3):1033–1066, 2008. Special Issue on Tensor Decompositions and Applications.

[34] L. De Lathauwer and J. Castaing. Tensor-based techniques for the blind separation of DS-CDMA signals. *Signal Processing*, 87(2):322–336, February 2007. Special Issue on Tensor Signal Processing.

[35] L. De Lathauwer, B. De Moor, and J. Vandewalle. An introduction to independent component analysis. *J. Chemometrics*, 14(3):123–149, May/June 2000. Special Issue: Multi-way Analysis.

[36] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, April 2000.

[37] L. De Lathauwer, B. De Moor, and J. Vandewalle. On the best rank-1 and rank-$(R_1, R_2, \ldots, R_N)$ approximation of higher-order tensors. *SIAM J. Matrix Anal. Appl.*, 21(4):1324–1342, April 2000.

[38] L. De Lathauwer, B. De Moor, and J. Vandewalle. Independent component analysis and (simultaneous) third-order tensor diagonalization. *IEEE Trans. Signal Process.*, 49(10):2262–2271, October 2001.

[39] L. De Lathauwer, B. De Moor, and J. Vandewalle. Computation of the canonical decomposition by means of a simultaneous generalized Schur decomposition. *SIAM J. Matrix Anal. Appl.*, 26(2):295–327, 2004.

[40] L. De Lathauwer and D. Nion. Decompositions of a higher-order tensor in block terms — Part III: Alternating least squares algorithms. *SIAM J. Matrix Anal. Appl.*, 30(3):1067–1083, 2008. Special Issue on Tensor Decompositions and Applications.

[41] L. De Lathauwer and J. Vandewalle. Dimensionality reduction in higher-order signal processing and rank-$(R_1, R_2, \ldots, R_N)$ reduction in multilinear algebra. *Linear Algebra Appl.*, 391:31–55, November 2004. Special Issue on Linear Algebra in Signal and Image Processing.

[42] V. de Silva and L.-H. Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM J. Matrix Anal. Appl.*, 30(3):1084–1127, 2008.

[43] M. De Vos, L. De Lathauwer, B. Vanrumste, S. Van Huffel, and W. Van Paesschen. Canonical decomposition of ictal scalp EEG and accurate source localization: Principles and simulation study. *Journal of Computational Intelligence and Neuroscience*, 2007(Article ID 58253):1–10, 2007. Special Issue on EEG/MEG Signal Processing.

[44] M. De Vos, A. Vergult, L. De Lathauwer, W. De Clercq, S. Van Huffel, P. Dupont, A. Palmini, and W. Van Paesschen. Canonical decomposition of ictal scalp EEG reliably detects the seizure onset zone. *NeuroImage*, 37(3):844–854, September 2007.

[45] J.-P. Dedieu, P. Priouret, and G. Malajovich. Newton's method on Riemannian manifolds: Covariant alpha theory. *IMA Journal of Numerical Analysis*, 23(3):395–419, 2003.

[46] J. Dehaene, M. Moonen, and J. Vandewalle. Analysis of a class of continuous-time algorithms for principal component analysis and subspace tracking. *IEEE Trans. Circuits Systems I Fund. Theory Appl.*, 46(3):364–372, 1999.

[47] J. E. Dennis, Jr. and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. Prentice Hall Series in Computational Mathematics. Prentice Hall Inc., Englewood Cliffs, NJ, 1983.

[48] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20(2):303–353, 1998.

[49] S. Eisenstat and H. Walker. Choosing the forcing terms in an inexact Newton method. *SIAM J. Sci. Comput.*, 17(1):16–32, 1996.

[50] M. Elad, P. Milanfar, and G. H. Golub. Shape from moments — an estimation theory perspective. *IEEE Transactions on Signal Processing*, 52(7):1814–1829, July 2004.

[51] L. Eldén and B. Savas. A Newton–Grassmann method for computing the best multi-linear rank-(r_1, r_2, r_3) approximation of a tensor. *SIAM J. Matrix Anal. Appl.*, 31(2):248–271, 2009.

[52] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *Comput. J.*, 7:149–154, 1964.

[53] K. R. Fowler and C. T. Kelley. Pseudo-transient continuation for nonsmooth nonlinear equations. *SIAM J. Numer. Anal.*, 43(4):1385–1406 (electronic), 2005.

[54] D. Gabay. Minimizing a differentiable function over a differential manifold. *J. Optim. Theory Appl.*, 37(2):177–219, 1982.

[55] F. R. Gantmacher. *The Theory of Matrices I, II*. Chelsea, New-York, 1959.

[56] J. C. Gilbert and J. Nocedal. Global convergence properties of conjugate gradient methods for optimization. *SIAM J. Optim.*, 2(1):21–42, 1992.

[57] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, Maryland, 3rd edition, 1996.

[58] M. Haardt, F. Roemer, and G. Del Galdo. Higher-order SVD-based subspace estimation to improve the parameter estimation accuracy in multidimensional harmonic retrieval problems. *IEEE Transactions on Signal Processing*, 56(7):3198–3213, 2008.

[59] W. Hackbusch and B. Khoromskij. Tensor-product approximation to multi-dimensional integral operators and Green's functions. *SIAM J. Matrix Anal. Appl.*, 30(3):1233–1253, 2008. Special Issue on Tensor Decompositions and Applications.

[60] W. W. Hager and H. Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization*, 16(1):170–192, 2005.

[61] R. A. Harshman. Foundations of the PARAFAC procedure: Model and conditions for an "explanatory" multi-mode factor analysis. *UCLA Working Papers in Phonetics*, 16(1):1–84, 1970.

[62] U. Helmke and J. B. Moore. *Optimization and Dynamical Systems*. Springer-Verlag, 1993.

[63] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436 (1953), 1952.

[64] D. J. Higham. Trust region algorithms and timestep selection. *SIAM J. Numer. Anal.*, 37(1):194–210 (electronic), 1999.

[65] F. L. Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematical Physics*, 6(1):164–189, 1927.

[66] F. L. Hitchcock. Multiple invariants and generalized rank of a *p*-way matrix or tensor. *Journal of Mathematical Physics*, 7(1):39–79, 1927.

[67] M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer. Best low multilinear rank approximation with conjugate gradients. Technical Report 09-246, ESAT-SISTA, K.U.Leuven, Belgium, 2009.

[68] M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer. Local minima of the best low multilinear rank approximation of higher-order tensors. Technical Report 09-247, ESAT-SISTA, K.U.Leuven, Belgium, 2009.

[69] M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. The best rank-(R_1, R_2, R_3) approximation of tensors by means of a geometric Newton method. In T. E. Simos, editor, *Proc. of the 6th International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2008), Kos, Greece*, pages 274–277, September 2008.

[70] M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. Dimensionality reduction for higher-order tensors: algorithms and applications. *International Journal of Pure and Applied Mathematics*, 42(3):337–343, 2008.

[71] M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme. Technical Report 09-142, ESAT-SISTA, K.U.Leuven, Belgium, 2009.

[72] M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. Differential-geometric Newton method for the best rank-(R_1, R_2, R_3) approximation of tensors. *Numerical Algorithms*, 51(2):179–194, June 2009. Tributes to Gene H. Golub Part II.

[73] M. Ishteva, L. De Lathauwer, and S. Van Huffel. Comparison of the performance of matrix and tensor based multi-channel harmonic analysis. In *7th International Conference on Mathematics in Signal Processing, Cirencester, UK*, pages 77–80, December 2006.

[74] M. V. Jankovic and H. Ogawa. Modulated Hebb–Oja learning rule—a method for principal subspace analysis. *IEEE Trans. Neural Networks*, 17(2):345–356, 2006.

[75] J. Jensen. *Sinusoidal Models for Speech Signal Processing*. PhD thesis, Aalborg University, Denmark, August 2000.

[76] M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization for semidefinite convex problems. Technical Report arXiv:0807.4423, 2008. http://arxiv.org/abs/0807.4423.

[77] C. T. Kelley, L.-Z. Liao, L. Qi, M. T. Chu, J. P. Reese, and C. Winton. Projected pseudo-transient continuation. *SIAM J. Numer. Anal.*, 46(6):3071–3083, 2008.

[78] B. N. Khoromskij and V. Khoromskaia. Low rank Tucker-type tensor approximation to classical potentials. *Central European Journal of Mathematics*, 5(3):523–550, 2007.

[79] S. Kobayashi and K. Nomizu. *Foundations of Differential Geometry*. Interscience Publishers, a division of John Wiley & Sons, New York-London, 1963. Volumes 1 and 2.

[80] E. Kofidis and P. A. Regalia. On the best rank-1 approximation of higher-order supersymmetric tensors. *SIAM J. Matrix Anal. Appl*, 23(3):863–884, 2002.

[81] T. Kolda. Orthogonal tensor decompositions. *SIAM J. Matrix Anal. Appl.*, 23:243–255, 2001.

[82] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, September 2009.

[83] P. M. Kroonenberg. *Applied Multiway Data Analysis*. Wiley, 2008.

[84] P. M. Kroonenberg and J. de Leeuw. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45(1):69–97, 1980.

[85] J. B. Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95–138, 1977.

[86] J. B. Kruskal. Rank, decomposition, and uniqueness for 3-way and n-way arrays. In R. Coppi and S. Bolasco, editors, *Multiway data analysis*, pages 7–18. North-Holland Publishing Co., Amsterdam, The Netherlands, 1989.

[87] J. M. Lee. *Introduction to smooth manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2003.

[88] C. Li and J. Wang. Newton's method on Riemannian manifolds: Smale's point estimate theory under the γ-condition. *IMA J. Numer. Anal.*, 26(2):228–251, 2006.

[89] E. Lundström and L. Eldén. Adaptive eigenvalue computations using Newton's method on the Grassmann manifold. *SIAM J. Matrix Anal. Appl.*, 23(3):819–839, 2001/02.

[90] X.-L. Luo, C. T. Kelley, L.-Z. Liao, and H.-W. Tam. Combining trust region techniques and Rosenbrock methods for gradient systems. *Journal of Optimization Theory and Applications*, 140(2):265–286, February 2009.

[91] M. W. Mahoney, M. Maggioni, and P. Drineas. Tensor-CUR decompositions for tensor-based data. *SIAM J. Matrix Anal. Appl.*, 30(3):957–987, 2008. Special Issue on Tensor Decompositions and Applications.

[92] R. Mahony and J. H. Manton. The geometry of the Newton method on non-compact Lie groups. *J. Global Optim.*, 23(3-4):309–327, 2002. Nonconvex optimization in control.

[93] J. H. Manton. Optimization algorithms exploiting unitary constraints. *IEEE Trans. Signal Process.*, 50(3):635–650, 2002.

[94] J. H. Manton, U. Helmke, and I. M. Y. Mareels. A dual purpose principal and minor component flow. *Systems Control Lett.*, 54(8):759–769, 2005.

[95] P. McCullagh. *Tensor Methods in Statistics*. Chapman and Hall, London, 1987.

[96] C. D. Moravitz Martin and C. F. Van Loan. A Jacobi-type method for computing orthogonal tensor decompositions. *SIAM J. Matrix Anal. Appl.*, 30(3):1219–1232, 2008. Special Issue on Tensor Decompositions and Applications.

[97] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM J. Sci. Statist. Comput.*, 4(3):553–572, 1983.

[98] G. Morren, P. Lemmerling, and S. Van Huffel. Decimative subspace-based parameter estimation techniques. *Signal Process.*, 83(5):1025–1033, 2003.

[99] M. Mørup, L. K. Hansen, C. S. Herrmann, J. Parnas, and S. M. Arnfred. Parallel factor analysis as an exploratory tool for wavelet transformed event-related EEG. *NeuroImage*, 29:938–947, 2006.

[100] C. L. Nikias and J. M. Mendel. Signal processing with higher-order spectra. *IEEE Signal Process. Mag.*, 10:10–37, 1993.

[101] C. L. Nikias and A. Petropulu. *Higher-Order Spectra Analysis. A Nonlinear Signal Processing Framework*. Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[102] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Verlag, New York, 2nd edition, 2006. Springer Series in Operations Research.

[103] E. Oja. A simplified neuron model as a principal component analyzer. *J. Math. Biol.*, 15:267–273, 1982.

[104] E. Oja. Neural networks, principal components, and subspaces. *Int. J. Neural Syst.*, 1:61–68, 1989.

[105] E. Oja, H. Ogawa, and J. Wangviwattana. Principal component analysis by homogeneous neural networks, Part I: The weighted subspace criterion. *IEICE Trans. Inf. Syst.*, 3:366–375, 1992.

[106] E. Oja, H. Ogawa, and J. Wangviwattana. Principal component analysis by homogeneous neural networks, Part II: Analysis and extensions of the learning algorithms. *IEICE Trans. Inf. Syst.*, 3:376–382, 1992.

[107] B. O'Neill. *Semi-Riemannian Geometry*, volume 103 of *Pure and Applied Mathematics*. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1983.

[108] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

[109] I. V. Oseledets, D. V. Savostianov, and E. E. Tyrtyshnikov. Tucker dimensionality reduction for three-dimensional arrays in linear time. *SIAM J. Matrix Anal. Appl.*, 30(3):939–956, 2008. Special Issue on Tensor Decompositions and Applications.

[110] J.-M. Papy. *Subspace-based Exponential Data Fitting Using Linear and Multilinear Algebra*. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, October 2005.

[111] J.-M. Papy, L. De Lathauwer, and S. Van Huffel. Exponential data fitting using multilinear algebra: The single-channel and the multichannel case. *Numer. Linear Algebra Appl.*, 12(8):809–826, October 2005.

[112] J.-M. Papy, L. De Lathauwer, and S. Van Huffel. Exponential data fitting using multilinear algebra: The decimative case. *J. Chemometrics*, 23(7–8):341–351, July/August 2009. Special Issue in Honor of Professor Richard A. Harshman.

[113] E. Polak and G. Ribière. Note sur la convergence de méthodes de directions conjuguées. *Rev. Française Informat. Recherche Opérationnelle*, 3(16):35–43, 1969.

[114] M. J. D. Powell. Some convergence properties of the conjugate gradient method. *Math. Programming*, 11(1):42–49, 1976.

[115] M. J. D. Powell. Restart procedures for the conjugate gradient method. *Math. Programming*, 12(2):241–254, 1977.

[116] M. J. D. Powell. Nonconvex minimization calculations and the conjugate gradient method. In *Numerical Analysis (Dundee, 1983)*, volume 1066 of *Lecture Notes in Math.*, pages 122–141. Springer, Berlin, 1984.

[117] B. Savas and L. Eldén. Krylov subspace methods for tensor computations. Technical Report LITH-MAT-R-2009-02-SE, Department of Mathematics, Linköping University, 2009.

[118] B. Savas and L.-H. Lim. Best multilinear rank approximation of tensors with quasi-Newton methods on Grassmannians. Technical Report LITH-MAT-R-2008-01-SE, Department of Mathematics, Linköping University, 2008.

[119] M. Shub. Some remarks on dynamical systems and numerical analysis. In L. Lara-Carrero and J. Lewowicz, editors, *Proc. VII ELAM.*, pages 69–92. Equinoccio, U. Simón Bolívar, Caracas, 1986.

[120] N. Sidiropoulos, R. Bro, and G. Giannakis. Parallel factor analysis in sensor array processing. *IEEE Trans. Signal Process.*, 48:2377–2388, 2000.

[121] N. Sidiropoulos, G. Giannakis, and R. Bro. Blind PARAFAC receivers for DS-CDMA systems. *IEEE Trans. Signal Process.*, 48:810–823, 2000.

[122] N. D. Sidiropoulos and R. Bro. On the uniqueness of multilinear decomposition of N-way arrays. *J. Chemometrics*, 14(3):229–239, June 2000. Special Issue: Multiway Analysis.

[123] A. Smilde, R. Bro, and P. Geladi. *Multi-way Analysis. Applications in the Chemical Sciences*. John Wiley and Sons, Chichester, U.K., 2004.

[124] S. T. Smith. *Geometric Optimization Methods for Adaptive Filtering*. PhD thesis, Division of Applied Sciences, Harvard University, Cambridge, MA, May 1993.

[125] S. T. Smith. Optimization techniques on Riemannian manifolds. In A. Bloch, editor, *Hamiltonian and gradient flows, algorithms and control*, volume 3 of *Fields Inst. Commun.*, pages 113–136. Amer. Math. Soc., Providence, RI, 1994.

[126] A. Stegeman and N. D. Sidiropoulos. On kruskal's uniqueness condition for the candecomp/parafac decomposition. *Linear Algebra and its Applications*, 420(2–3):540–552, 2007.

[127] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20(3):626–637, 1983.

[128] P. Stoica and R. Moses. *Spectral Analysis of Signals*. Pearson, Prentice Hall, 2005.

[129] P. Stoica and A. E. Nordsjö. Subspace-based frequency estimation in the presence of moving-average noise using decimation. *Signal Process.*, 63(3):211–220, 1997.

[130] A. Swami and G. Giannakis. Bibliography on HOS. *Signal Process.*, 60:65–126, 1997.

[131] J. ten Berge, J. de Leeuw, and P. Kroonenberg. Some additional results on principal components analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 52(2):183–191, June 1987.

[132] P. L. Toint. Towards an efficient sparsity exploiting Newton method for minimization. In I. S. Duff, editor, *Sparse Matrices and Their Uses*, pages 57–88. Academic Press, London, 1981.

[133] L. R. Tucker. The extension of factor analysis to three-dimensional matrices. In H. Gulliksen and N. Frederiksen, editors, *Contributions to mathematical psychology*, pages 109–127. Holt, Rinehart & Winston, NY, 1964.

[134] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966.

[135] C. Udrişte. *Convex functions and optimization methods on Riemannian manifolds*, volume 297 of *Mathematics and its Applications*. Kluwer Academic Publishers Group, Dordrecht, 1994.

[136] S. Van Huffel, H. Chen, C. Decanniere, and P. Van Hecke. Algorithm for time-domain NMR data fitting based on total least squares. *J. Magn. Reson. A*, 110:228–237, 1994.

[137] L. Vanhamme. *Advanced time-domain methods for nuclear magnetic resonance spectroscopy data analysis*. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, November 1999.

[138] L. Vanhamme, T. Sundin, P. Van Hecke, and S. Van Huffel. MR spectroscopy quantitation: a review of time-domain methods. *NMR Biomed.*, 14(4):233–46, 2001.

[139] L. Vanhamme and S. Van Huffel. Multichannel quantification of biomedical magnetic resonance spectroscopy signals. In F. Luk, editor, *Proc. of SPIE, Advanced Signal Processing Algorithms, Architectures, and Implementations VIII*, volume 3461, pages 237–248, San Diego, California, July 1998.

[140] M. A. O. Vasilescu and D. Terzopoulos. Multilinear subspace analysis for image ensembles. In *Proc. Computer Vision and Pattern Recognition Conf. (CVPR '03), Madison, WI*, volume 2, pages 93–99, 2003.

[141] L. Xu. Least mean square error reconstruction principle for self-organizing neural-nets. *Neural Networks*, 6(5):627–648, 1993.

[142] W.-Y. Yan. Stability and convergence of principal component learning algorithms. *SIAM J. Matrix Anal. Appl.*, 19(4):933–955 (electronic), 1998.

[143] W.-Y. Yan, U. Helmke, and J. B. Moore. Global analysis of Oja's flow for neural networks. *IEEE Trans. on Neural Networks*, 5(5):674–683, 1994.

[144] S. Yoshizawa, U. Helmke, and K. Starkov. Convergence analysis for principal component flows. *Int. J. Appl. Math. Comput. Sci.*, 11(1):223–236, 2001.

[145] S. Yoshizawa, U. Helmke, and K. Starkow. Corrections to: "Convergence analysis for principal component flows" [Int. J. Appl. Math. Comput. Sci. **11** (2001), no. 1, 223–236; MR1835155 (2002d:93052)]. *Int. J. Appl. Math. Comput. Sci.*, 12(2):299, 2002.

[146] T. Zhang and G. H. Golub. Rank-one approximation to high order tensors. *SIAM J. Matrix Anal. Appl.*, 23:534–550, 2001.

Publication list

Journal papers

International

1. M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. Differential-geometric Newton method for the best rank-(R_1, R_2, R_3) approximation of tensors. *Numerical Algorithms*, 51(2):179–194, June 2009. Tributes to Gene H. Golub Part II.

2. P.-A. Absil, M. Ishteva, L. De Lathauwer, and S. Van Huffel. A geometric Newton method for Oja's vector field. *Neural Comput.*, 21(5):1415–1433, May 2009.

3. M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. Dimensionality reduction for higher-order tensors: algorithms and applications. *International Journal of Pure and Applied Mathematics*, 42(3):337–343, 2008.

4. M. Ishteva, L. Boyadjiev, and R. Scherer. On the Caputo operator of fractional calculus and C-Laguerre functions. *Mathematical Sciences Research Journal*, 9(6):161–170, June 2005.

National

5. M. Ishteva and L. Boyadjiev. On the C-Laguerre functions. *Comptes rendus de l'Academie bulgare des Sciences*, 58(9):1019–1024, 2005.

Conference Papers

International

6. M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. The best rank-(R_1, R_2, R_3) approximation of tensors by means of a geometric Newton method. In T. E. Simos, editor, *Proc. of the 6th International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2008), Kos, Greece*, pages 274–277, September 2008.
7. M. Ishteva, L. De Lathauwer, and S. Van Huffel. Comparison of the performance of matrix and tensor based multi-channel harmonic analysis. In *7th International Conference on Mathematics in Signal Processing, Cirencester, UK*, pages 77–80, December 2006.
8. N. Traykov, M. Ishteva, and A. Ichtev. Multiple model approach for fault detection and isolation. In *Proc. of the ICAMES'02, Istanbul, Turkey*, May 2002. Published in CD. Student conference.

National

9. Ph. Michaylov, P. Yordanov, M. Ishteva, Ph. Philipoff, and P. Georgieva. Satellite aerial automatical vectoring. In *Proc. of the XXV Summer School, "Applications of Mathematics in Engineering and Economics", Sozopol'99, Heron Press, Sofia*, pages 133–136, 2000.
10. K. Ishtev, Ph. Philipoff, M. Ishteva, and Ph. Michaylov. Structural hard transport vehicle modeling. In *Proc. of the XXIV Summer School, "Applications of Mathematics in Engineering", Sozopol'98, Heron Press, Sofia*, pages 175–178, 1999.
11. Ph. Philipoff, K. Ishtev, M. Ishteva, and Ph. Michaylov. Algorithms for dynamical condensation of structures' models. In *Proc. of the VVISU'98, Sofia*, pages 171–178, 1998. In Bulgarian.

Internal reports (including submitted papers)

12. M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel. Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme. Technical Report 09-142, ESAT-SISTA, K.U.Leuven, Belgium, 2009.
13. M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer. Best low multilinear rank approximation with conjugate gradients. Technical report 09-246, ESAT-SISTA, K.U.Leuven, Belgium, 2009.

14. M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer. Local minima of the best low multilinear rank approximation of higher-order tensors. Technical report 09-247, ESAT-SISTA, K.U.Leuven, Belgium, 2009.

Abstracts at conferences

15. M. Ishteva, L. De Lathauwer, P.-A. Absil, S. Van Huffel. Trust-region method for the best multilinear rank approximation of tensors. *Minisymposium on Multilinear Methods in Numerical Linear Algebra at the SIAM Conference on Applied Linear Algebra*, Oct. 26–29, Seaside, California; invited by Lek-Heng Lim (University of California, Berkeley) and E. Tyrtyshnikov (Moscow State Univ.).

16. M. Ishteva, L. De Lathauwer, P.-A. Absil, S. Van Huffel. Best multilinear rank approximation of tensors: A trust-region algorithm and local minima. *14th Belgian-French-German Conference on Optimization*, Sep. 14–18, 2009, Leuven, Belgium.

17. M. Ishteva, L. De Lathauwer, P.-A. Absil, S. Van Huffel. Best low multilinear rank approximation of tensors and its local minima. *European Workshop on Challenges in Modern Massive Data Sets (EMMDS 2009)*, July 1–4, 2009, Copenhagen, Denmark.

18. L. De Lathauwer, M. Ishteva, P.-A. Absil, S. Van Huffel. Best multilinear rank approximation: New results. *ThRee-way methods In Chemistry And Psychology (TRICAP 2009)*, June 14–19, Vall de Núria, Spain; invited by R. Tauler (Inst. of Environmental Assessment and Water Research, Spain).

19. M. Ishteva, L. De Lathauwer, P.-A. Absil, S. Van Huffel. Differential-geometric Newton method for the best rank-$(R_1, R_2, \ldots, R_N)$ approximation of tensors. *80th Annual Meeting of the International Association of Applied Mathematics and Mechanics (GAMM 2009)*, Feb. 9–13, 2009, Gdansk, Poland; invited by L. Grasedyck (Max-Planck- Institute for Mathematics in the Sciences, Leipzig, Germany) and E. Tyrtyshnikov (Moscow State Univ.).

20. M. Ishteva, L. De Lathauwer, P.-A. Absil, S. Van Huffel. Numerical methods for the best multilinear rank approximation of higher-order tensors. *Workshop of Computational Polynomial Optimization and Multilinear Algebra*, Dec. 22–23, 2008, Hong Kong; invited by L. Qi (City Univ. of Hong Kong).

21. M. Ishteva, L. De Lathauwer, P.-A. Absil, S. Van Huffel. Higher-order tensors: Best rank-$(R_1, R_2, \ldots, R_N)$ approximation algorithms. *SIAM Conf. on Optimization (SIOPT)*, May 10–13, 2008, Boston, MA, U.S.A.; invited by K. Hüper (Australian National Univ., Canberra, Australia) and Prof. P.-A. Absil (UCL, Louvain-la-Neuve, Belgium).

22. M. Ishteva, L. De Lathauwer, P.-A. Absil, S. Van Huffel. Higher-order tensors: Best rank-$(R_1, R_2, \ldots, R_N)$ approximation algorithms. *Miniworkshop on Higher-Order Tensors*, Oct. 16, 2007, Dept. of Mathematics, Linköping Univ., Linköping, Sweden; invited by L. Eldén, (Linköping Univ., Linköping, Denmark).

23. M. Ishteva, L. De Lathauwer, S. Van Huffel. Dimensionality reduction for higher-order tensors: Algorithms and applications. *Fourth Int. Conf. of Applied Mathematics and Computing*, Aug. 12–18, 2007, Plovdiv, Bulgaria; invited by Prof. D. Bainov (Medical Univ. of Sofia, Sofia, Bulgaria).

24. M. Ishteva, L. De Lathauwer, P.-A. Absil, R. Sepulchre, S. Van Huffel. Best rank-(R_1, R_2, R_3) approximation of third-order tensors, based on the Riemannian trust-region scheme. *6th Int. Congress on Industrial and Applied Mathematics (ICIAM 2007)*, July 16–20, 2007, Zürich, Switzerland; invited by I. Ibragimov (Univ. Saarland, Germany).

25. M. Ishteva, L. De Lathauwer, P.-A. Absil, R. Sepulchre, S. Van Huffel. Rank-(R_1, R_2, R_3) reduction of tensors based on the Riemannian trust-region scheme. *Fourth Total Least Squares and Errors-in-Variables Modeling Workshop*, Aug. 21–23, 2006, Leuven, Belgium.

26. M. Ishteva, L. De Lathauwer, P.-A. Absil, R. Sepulchre, S. Van Huffel. Rank-(R_1, R_2, R_3) reduction of tensors based on the Riemannian trust-region scheme. *Workshop on Algorithms for Modern Massive Data Sets*, June 21–24, Stanford University and Yahoo! Research, 2006.

Other scientific meetings

27. M. Ishteva, M. De Vos, L. De Lathauwer, P.-A. Absil, R. Sepulchre, S. Van Huffel. Numerical algorithms for tensor decompositions and applications. *Kickoff meeting of the IAP network VI/4 - DYSCO*, Apr. 16, 2007, Louvain-la-Neuve.

28. M. Ishteva, L. De Lathauwer, S. Van Huffel. Tensor based multi-channel harmonic analysis. *Study day of the Interuniversity Attraction Pole IAP V/22*, May 16, 2006, Louvain-la-Neuve.

Award

- *Best poster from a didactical point of view*, 14th Belgian-French-German Conference on Optimization, Sep. 14–18, 2009, Leuven, Belgium.

Curriculum vitae

Mariya Ishteva was born on April 15, 1980 in Sofia, Bulgaria. She received the Bachelor's degree in Computer Science from the Faculty of Mathematics and Computer Science, Sofia University in July 2002. She continued at the Faculty of Mathematics, Universität Karlsruhe (TH), Germany, first as an ERASMUS student and later as a master student. In February 2005, she obtained the Master's degree in Mathematics. The topic of her Master's thesis was Properties and applications of the Caputo fractional operator.

She started research at ESAT/SCD, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium in October 2005 as a predoctoral student and continued with her PhD studies in January 2006. Her research was supported by

(1) K.U.Leuven predoctoral scholarship; K.U.Leuven doctoral scholarships in the framework of the Central- and Eastern-European initiatives (OE/06/25, OE/07/17, OE/08/007, OE/09/004),
(2) Research Council K.U.Leuven: GOA-Ambiorics, CoE EF/05/006 Optimization in Engineering (OPTEC),
(3) F.W.O. project G.0321.06, "Numerical tensor methods for spectral analysis",
(4) the Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, "Dynamical systems, control and optimization", 2007–2011).

Arenberg Doctoral School of Science, Engineering & Technology
Faculty of Engineering
Department of Electrical Engineering
Research group ESAT/SCD
Address Kasteelpark Arenberg 10/2446, B-3001 Leuven (Belgium)

KATHOLIEKE UNIVERSITEIT LEUVEN

ASSOCIATIE K.U.LEUVEN