KU LEUVEN

ARENBERG DOCTORAL SCHOOL
FACULTY OF ENGINEERING SCIENCE

Explicit and implicit tensor decomposition-based algorithms and applications

Martijn Boussé

Supervisor:
Prof. dr. ir. L. De Lathauwer

Dissertation presented in partial
fulfillment of the requirements for the
degree of Doctor of Engineering
Science (PhD): Electrical Engineering

September 2019

Explicit and implicit tensor decomposition-based algorithms and applications

Martijn BOUSSÉ

Examination committee:
Prof. dr. A. Bultheel, chair
Prof. dr. ir. L. De Lathauwer, supervisor
Prof. dr. ir. K. Meerbergen
Prof. dr. ir. T. van Waterschoot
Prof. dr. ir. S. Van Huffel
Prof. dr. N. Sidiropoulos
(University of Virginia, USA)
Prof. dr. ir. M. Verhaegen
(TU Delft, The Netherlands)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Electrical Engineering

September 2019

© 2019 KU Leuven – Faculty of Engineering Science
Uitgegeven in eigen beheer, Martijn Boussé, Kasteelpark Arenberg 10 bus 2440, B-3001 Leuven (Belgium).

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, 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, electronic or any other means without written permission from the publisher.

Preface

First of all, I would like to thank my promotor, prof. Lieven De Lathauwer, for giving me the opportunity to be part of his team after I successfully finished my master thesis under his supervision. While it was not always easy to process his elaborate feedback and detailed questions, I must admit that it has helped me to become a better researcher. The proof can be seen in the enormous improvements in my scientific output over the last five years. As a famous psychologist once said: “No tree can grow to heaven unless its roots reach down to hell.” Lieven, I believe our collaboration has been fruitful and your support, expertise, and experience has had a huge impact on that.

I would also like to thank the other members of my examination committee, Chairman Prof. Adhemar Bultheel, Prof. Karl Meerbergen, Prof. Toon van Waterschoot, Prof. Sabine Van Huffel, Prof. Nikos Sidiropoulos, and Prof. Michel Verhaegen, for the feedback and comments during the preliminary defense. Karl and Toon, thank you for your valuable input during the course of my PhD and at the intermediate presentations. Sabine, I greatly appreciated the various collaborations with you and your team during the Biotensors project. Nikos, thank you for welcoming me in your team during my research stay in the US, which I believe has led to a productive collaboration.

My research has received funding from the European Research Council under the European Union’s Seventh Framework Programme (FP7/2007-2013) / ERC Advanced Grant: BIOTENSORS (no. 339804), Fonds de la Recherche Scientifique – FNRS and Fonds Wetenschappelijk Onderzoek – Vlaanderen under EOS Project no. 30468160 (SeLMA), and Research Council KU Leuven: C1 project C16/15/059-nD. In 2018, I was a visiting researcher at the University of Virginia, Charlottesville, and University of Minnesota, Minneapolis with prof. Nikos Sidiropoulos, which was supported by a travel grant from the Flanders Research Foundation (FWO). I would also like to thank FWO for supporting my visit to the GlobalSIP conference in 2018.

During my PhD, I have had the opportunity to travel to many interesting, and less interesting, conferences. Besides, the United Kingdom, France, Germany, Spain, Italy, and Hungary, I’ve also visited more exotic conference locations of which I have fond memories such as drinking cocktails on the beach with other researchers and professors at the Santa Barbara Resort in Curaçao. Another unforgetable conference was at Jeju Island in South Korea with Amir and Mario. I’ll never forget Amir’s dedication to research when he handed his scientific poster to Mario and said: “Present the research!” while he was split from us at Chinese customs. Another worthwhile anecdote: I climbed the Hallasan of Jeju island and sitting there on top of that dormant volcano, I decided to propose to my girlfriend. (Flash forward a year and I actually proposed to her on another volcano, but this time an active one in Indonesia for additional suspense.) Less exotic, but still magical, was my trip to Anaheim, also known as the location of the original Disneyland Park and, coincidentally, the location of the conference that I was attending.

In the fourth year of my PhD, I received a grant from FWO for a long research stay with Prof. Nikos Sidiropoulos at the University of Minnesota and the University of Virginia. When I arrived in Minneapolis, I had the pleasure of meeting an actual snow storm in which my face almost froze off while walking to my AirBnB in a sketchy neighbourhood. Good times! During my stay in Minneapolis I hung out a few times with the Greek gang — for some reason, there were a lot of Greek PhD students in Minneapolis — and we visited the frozen waterfall in the beautiful Minnehaha park, we drove around the frozen lakes, and we had some ice cream. The ice cream actually became cheaper with decreasing temperatures so it was really cheap. In retrospect, my visit to the University of Minnesota and the University of Virginia where one of the highlights of my PhD. Looking back I would like to have stayed longer and do more. I also really enjoyed my time with the people from Nikos’ team; we’ve had some great conversations and dinners.

A friend of mine once told me that one of the pillars of a successful PhD are collaborations with other researchers and especially researchers that are active in different domains. Through the Biotensors project, I was granted a unique position to collaborate with talented biomedical engineers, allowing me to apply my techniques to real-world applications. I want to highlight my collaboration with Griet on irregular heartbeat classification which has sparked several other collaborations with Simon, another Simon, and Siba, leading to conference papers and even a book chapter. I also enjoyed my collaborations with Otto, Nico, Ignat, and Michiel on various papers.

I have had a great time at the ESAT office, especially with Frederik, Otto, and Nico in the earlier years of my PhD. Frederik and I were in the same year of our Master in Mathematical Engineering and we worked together on several projects, which I really enjoyed. Later on we became colleagues during our PhD and I have many fond memories of our daily office talk. Otto was my master thesis supervisor and later became one of the colleagues with which I have worked very closely. Thank you Otto for being such a positive influence on me during my PhD. You really have played a great part in making me the researcher and the person I am today. Nico, I appreciate your help and guidance in making me a better researcher and to think more quickly on my feet. I also want to thank you explicitly for all the Latex and Tensorlab support, especially in the final year. Besides the colleagues in Leuven, I also want to thank the colleagues at Kortrijk for being such great hosts whenever I visited the faraway and strange lands of West-Vlaanderen. Thank you Michiel, Ignat, Mikael, Alwin, Patrick, Xiaofeng, Chuan, and Geunseop. Also, thank you Dries, Erik, Benjamin, and Kevin, for the five wonderful years of supervising P&O2 at Kulak!

Even though I was not always present at the various and numerous Biomed events, I did really appreciate my time in the new ESAT building with the biomedical engineers. Thank you all for adopting the tensor team and letting us be part of Christmas parties, pool parties, and all other events (i.e., other parties). I'll never forget the high-stakes discussions about relevant, and less relevant, societal issues. Rob, thank you for the great parties and the great talks! I really look forward to our future collaborations. I also want to thank Philippe Dreesen from the VUB for some good talks in Curaçao and some interesting email discussions. Thank you Thomas for our tensor talks during lunch at ESAT or IMEC. I also want to thank Elsy, Ida, Jacqueline, John, Wim, and Maarten for all the help with the various administrative tasks.

I want to thank all of my friends and family members for supporting me and especially for listening to my rants about tensors and my amazing research. Kristof and Kevin, thank you for being there for me in good and bad times; I really appreciate both of you and I'm happy that I can call you my friends. Dario, thank you for being my personal fan boy, but I'm still hoping for a job opening in your future company. Daniel, thank you for telling me to be less critical of myself, especially in the early years of my PhD. A special thank you also goes to my parents who have laid the foundation for the person that I have become today. Ma and pa, I want to emphasize how much I appreciate all the opportunities that you have given me. Thank you for your support. Last but not least, I want to thank my wife, Jishi, for all the loving encouragement, patience and support that you have given me. It goes without a doubt that I would have never been able to finish this PhD without you. You truly are the love of my life and I'm so happy to share these moments with you. *Ad astra, per aspera.*

Martijn Boussé
September 2019

Abstract

Various real-life data such as time series and multi-sensor recordings can be represented by vectors and matrices, which are one-way and two-way arrays of numerical values, respectively. Valuable information can be extracted from these measured data matrices by means of matrix factorizations in a broad range of applications within signal processing, data mining, and machine learning. While matrix-based methods are powerful and well-known tools for various applications, they are limited to single-mode variations, making them ill-suited to tackle *multi-way data* without loss of information. Higher-order tensors are a natural extension of vectors (first order) and matrices (second-order), enabling us to represent multi-way arrays of numerical values, which have become ubiquitous in signal processing and data mining applications. By leveraging the powerful utitilies offered by tensor decompositions such as compression and uniqueness properties, we can extract more information from multi-way data than what is possible by using only matrix tools.

While higher-order tensors allow us to properly accommodate for multiple modes of variation in data, tensor problems are often large-scale because the number of entries in a tensor increases exponentially with the tensor order. This curse of dimensionality can, however, be alleviated or even broken by various techniques such as representing the tensor by an approximate but compact tensor model. While a pessimist only sees the curse, an optimist sees a significant opportunity for the compact representation of large-scale data vectors: by representing a large-scale vector (first order) using a compact (higher-order) tensor model, the number of parameters needed to represent the underlying vector decreases exponentially in the order of the tensor representation. The key assumption to employ this *blessing of dimensionality* is that the data can be described by much fewer parameters than the actual number of samples, which is often true in large-scale applications.

By leveraging the blessing of dimensionality in this thesis for blind source separation and (blind) system identification, we can tackle large-scale applications through *explicit and implicit tensor decomposition-based methods*. While explicit decompositions decompose a tensor that is known a priori, implicit decompositions decompose a tensor that is only known implicitly. In this thesis, we present a single-step framework for a particular type of implicit tensor decomposition, consisting of optimization-based and algebraic algorithms as well as generic uniqueness results. By properly exploiting additional structure in specific applications, we can significantly reduce the computational complexity of our optimization-based method. Our approach for large-scale instantaneous blind source separation and (blind) system identification enables various applications such as direction-of-arrival estimation in large-scale arrays and neural spike sorting in high-density recordings. Furthermore, we link implicit tensor decompositions to multilinear systems of equations, which are a generalization of linear systems, allowing us to propose a novel tensor-based classification scheme that we use for face recognition and irregular heartbeat classification with excellent performance.

Beknopte samenvatting

Vectoren en matrices zijn respectievelijk één- en tweewegse tabellen van getallen die gebruikt kunnen worden om allerlei soorten data zoals tijdsreeksen en multi-sensoropnames voor te stellen. Met behulp van matrixontbindingen kan men waardevolle informatie ontginnen uit dergelijke datamatrices in een brede waaier van toepassingen binnen signaalverwerking, dataontginning, en machinaal leren. Hoewel matrixtechnieken krachtige instrumenten zijn in heel wat toepassingen, zijn matrices ontoereikend om *meerwegsdata* voor te stellen zonder informatieverlies. Een hogere-ordetensor is een uitbreiding van een vector (eerste orde) en een matrix (tweede orde) die ons toelaat om de alomtegenwoordige meerwegsdata in signaalverwerking en dataontginning op een natuurlijke wijze voor te stellen. Door gebruik te maken van de krachtige eigenschappen van tensorontbindingen zoals compressie en uniciteitsvoorwaarden, kunnen we de resultaten van methoden die enkel matrices gebruiken overtreffen.

Terwijl hogere-ordetensoren uitermate geschikt zijn om meerwegsdata op een gepaste wijze te kunnen voorstellen, zijn tensorproblemen vaak grootschalig omdat het aantal waarden in een tensor exponentieel toeneemt met de orde van de tensor. Deze vloek van de dimensionaliteit kan echter verlicht of zelfs gebroken worden door allerlei technieken. Zo worden bijvoorbeeld tensorontbindingen gebruikt voor een compacte voorstelling van een tensor. Terwijl een pessimist enkel een vloek ziet, herkent een optimist een belangrijke opportuniteit voor de compacte voorstelling van grootschalige datavectoren: door de grootschalige vector (eerste orde) voor te stellen met een compact (hogere-orde) tensormodel, bekomen we een voorstelling waarvan het aantal parameters exponentieel afneemt met de orde van de tensor. Dit noemen we de *zegen van de dimensionaliteit*. Hierbij maken we de belangrijke veronderstelling dat de data kunnen voorgesteld worden door veel minder parameters dan het effectieve aantal bemonsteringen, wat vaak het geval is in grootschalige toepassingen.

Door gebruik te maken van de zegen van de dimensionaliteit in blinde signaalscheiding en (blinde) systeemidentificatie, kunnen we grootschalige toepassingen aanpakken via *ontbindingen van expliciet en impliciet gegeven tensoren*. Terwijl een expliciete ontbinding een tensor ontbindt die a priori gegeven is, ontbindt een impliciete ontbinding een tensor die enkel op impliciete wijze gekend is. We ontwikkelen in deze thesis generische uniciteitsvoorwaarden en een éénstapsraamwerk voor een bepaald type ontbinding van een impliciet gegeven tensor met behulp van optimalisatie en klassieke lineaire algebra. Door op gepaste wijze bijkomende structuur uit te buiten in specifieke toepassingen, kunnen we de computationele vereisten van onze optimalisatiegebaseerde algoritmen significant verlagen. Onze methode voor grootschalige ogenblikkelijke blinde signaalscheiding en (blinde) systeemidentificatie laat toe om allerlei toepassingen efficiënt aan te pakken. Voorbeelden zijn de schatting van de aankomstrichting van signalen die opgemeten worden door grote rijen van antennes alsook het scheiden van neurale pieken in hogedensiteitsopnames. We tonen verder aan dat de ontbinding van een impliciet gegeven tensor kan gerelateerd worden aan multilineaire stelsels van vergelijkingen, welke een veralgemening zijn van klassieke lineare stelsels. Dit laat ons toe om een nieuwe tensorgebaseerde classificatiemethode te ontwikkelen waarmee we gezichtsherkenning en irreguliere hartslagclassificatie nauwkeurig kunnen uitvoeren.

Contents

List of figures xv

List of tables xxi

List of algorithms xxiii

List of acronyms xxv

List of symbols xxvii

I Introduction 1

1 Introduction 3
- 1.1 From two-way to multi-way data . . . 3
- 1.2 From a curse to a blessing . . . 6
- 1.3 From explicit to implicit decompositions . . . 10
- 1.4 Research objectives . . . 10
- 1.5 Thesis overview . . . 13

2 Tensor decompositions 19
- 2.1 Introduction . . . 19
- 2.2 Basic definitions and notations . . . 20
  - 2.2.1 Notations . . . 20
  - 2.2.2 Terminology . . . 20
  - 2.2.3 Products . . . 20
  - 2.2.4 Rank . . . 22
  - 2.2.5 Properties . . . 23
- 2.3 Tensor decompositions . . . 23
  - 2.3.1 Multilinear singular value decomposition (MLSVD) . . . 23
  - 2.3.2 Canonical polyadic decomposition (CPD) . . . 24
  - 2.3.3 Block term decomposition (BTD) . . . 26
- 2.4 *All-at-once* optimization-based algorithms . . . 27

II Algorithms 29

3 Linear systems with a canonical polyadic decomposition constrained solution: algorithms and applications 31
3.1 Introduction . . . 32
3.1.1 Notation and definitions . . . 35
3.1.2 Multilinear algebraic prerequisites . . . 35
3.2 Linear systems with a CPD constrained solution . . . 36
3.2.1 Definition . . . 36
3.2.2 LS-CPD as CPD by exploiting structure of $\mathbf{A}$ . . . 38
3.2.3 Generic uniqueness . . . 39
3.3 Algorithms . . . 39
3.3.1 Algebraic computation . . . 40
3.3.2 Optimization-based methods . . . 42
3.4 Numerical experiments . . . 46
3.4.1 Proof-of-concept . . . 47
3.4.2 Comparison of methods . . . 48
3.4.3 Initialization methods . . . 49
3.4.4 Preconditioner . . . 50
3.5 Applications . . . 51
3.5.1 Tensor-based face recognition using LS-CPDs . . . 51
3.5.2 Constructing a tensor that has particular multilinear singular values . . . 53
3.5.3 Blind deconvolution of constant modulus signals . . . 55
3.6 Conclusion and future research . . . 56

4 NLS algorithm for Kronecker-structured linear systems with a CPD constrained solution 59
4.1 Introduction . . . 60
4.1.1 Notations and basic definitions . . . 61
4.1.2 Canonical Polyadic Decomposition (CPD) . . . 61
4.1.3 Identities and derivatives . . . 61
4.2 Kronecker-structured LS-CPD as a sum of CPDs with linearly constrained factor matrices . . . 62
4.3 Nonlinear least-squares (NLS) algorithm . . . 63
4.3.1 Objective function . . . 65
4.3.2 Jacobian . . . 65
4.3.3 Gradient . . . 65
4.3.4 Gramian-vector product . . . 66
4.3.5 Block-Jacobi preconditioner . . . 67
4.3.6 Computational complexity . . . 67
4.4 Experiments . . . 68
4.4.1 The block-Jacobi preconditioner is effective . . . 68
4.4.2 Graph clustering as a Kronecker-structured LS-CPD . . . 68

4.5 Conclusion 69

III Applications 71

5 A tensor-based method for large-scale blind source separation using segmentation 73
5.1 Introduction 74
5.1.1 Notation and definitions 75
5.1.2 Tensor decompositions 76
5.2 Large-scale blind source separation via low-rank sources . . . 78
5.2.1 Blind source separation 78
5.2.2 Low-rank sources 79
5.2.3 Decomposition 84
5.3 Large-scale blind source separation via low-rank mixing vectors 86
5.4 Large-scale blind source separation using twofold segmentation 87
5.5 Simulations and applications 89
5.5.1 General experiments 90
5.5.2 Underdetermined mixture 92
5.5.3 Noise and sample length 92
5.5.4 Low-rank approximation 93
5.5.5 Compression versus accuracy 94
5.5.6 Consistency 96
5.5.7 Fetal electrocardiogram extraction 98
5.5.8 Direction of arrival estimation 99
5.6 Conclusion 103

6 Tensor-based large-scale blind system identification using segmentation 105
6.1 Introduction 106
6.1.1 Notation and definitions 107
6.1.2 Tensor decompositions 108
6.2 Decomposition in (rank-$P_r \otimes$ vector) terms 109
6.2.1 Definition 110
6.2.2 Uniqueness 110
6.2.3 Algebraic method 112
6.3 Large-scale BSI using segmentation 113
6.3.1 Blind system identification 113
6.3.2 Low-rank coefficient vectors 114
6.3.3 Segmentation and decomposition 120
6.3.4 Uniqueness 122
6.3.5 Block-Toeplitz structure 123
6.4 Numerical experiments 127
6.4.1 Proof-of-concept 128

6.4.2 Influence of noise and sample size . . . 128
6.4.3 Influence of the system order L . . . 129
6.4.4 Parameter selection . . . 130
6.5 Applications . . . 132
6.5.1 Direction-of-arrival estimation . . . 132
6.5.2 Neural spike sorting . . . 136
6.6 Conclusion . . . 137

7 Large-scale autoregressive system identification using Kronecker product equations **139**
7.1 Introduction . . . 140
7.1.1 Notations and basic definitions . . . 141
7.1.2 Kronecker product equation . . . 141
7.2 Large-scale SIMO autoregressive system identification using KPEs . . . 142
7.2.1 Autoregressive system identification . . . 142
7.2.2 Kronecker constrained system coefficients . . . 143
7.2.3 Large-scale AR system identification as a KPE . . . 144
7.3 Experiments . . . 144
7.3.1 Proof-of-concept experiment . . . 145
7.3.2 Influence of noise and sample size on the accuracy . . . 145
7.3.3 Influence of the system order on the accuracy . . . 146
7.4 Conclusion and future work . . . 147

8 Face recognition as a Kronecker product equation **149**
8.1 Introduction . . . 150
8.1.1 Notations and basic definitions . . . 151
8.1.2 Multilinear singular value decomposition . . . 151
8.1.3 (Coupled) Kronecker product equations . . . 151
8.2 Face recognition using KPEs . . . 152
8.2.1 Tensorization and MLSVD model . . . 152
8.2.2 Kronecker product equation . . . 153
8.2.3 Face recognition . . . 153
8.3 Numerical experiments . . . 154
8.3.1 Proof-of-concept . . . 154
8.3.2 Performance . . . 155
8.3.3 Improving performance through coupling . . . 156
8.3.4 Updating the database with a new person . . . 156
8.4 Conclusion . . . 158

9 Irregular heartbeat classification using Kronecker product equations **159**
9.1 Introduction . . . 160
9.1.1 Notation and definitions . . . 161

9.1.2 Multilinear singular value decomposition 161
9.1.3 Kronecker product equations 162
9.2 Irregular heartbeat classification as a Kronecker product equation 162
9.2.1 Preprocessing and tensorization 162
9.2.2 Kronecker product equation 163
9.2.3 Irregular heartbeat classification 164
9.3 Results and discussion 165
9.4 Conclusion 167

10 Conclusion 169
10.1 Contributions 169
10.2 Perspectives 174

A Nonlinear least-squares algorithm for canonical polyadic decomposition using low-rank weights 177
A.1 Introduction 178
A.1.1 Notations and basic definitions 179
A.1.2 Canonical polyadic decomposition 179
A.1.3 Identities and derivatives 180
A.2 WLS for CPD using low-rank weights 180
A.2.1 Objective function 181
A.2.2 Jacobian 182
A.2.3 Gradient 183
A.2.4 Gramian-vector product 184
A.2.5 Block-Jacobi preconditioner 184
A.2.6 Complexity 185
A.3 Direction-of-arrival estimation using WLS 185
A.4 Conclusion 186

List of figures

1.1 While a vector (first order) and a matrix (second order) represent one-way and two-way arrays of data, resp., a higher-order tensor represents a multi-way array of data. 4

1.2 While tensors are by themselves already ubiquitous in a broad range of applications within signal processing, data mining, and machine learning, we also often want to construct a tensor from given data matrices. This is called *tensorization* and it allows us to leverage the powerful properties of higher-order tensors such as compression and uniqueness conditions. . . . 5

1.3 Matrix decompositions such as the well-known SVD can be generalized to the multilinear SVD, enabling compression and subspace analysis for higher-order tensors. 5

1.4 The number of entries in a higher-order tensor increases *exponentially* with the order of the tensor, which is known as the *curse of dimensionality.* 6

1.5 By approximating a tensorized representation of a large-scale data vector by a low-rank tensor decomposition, we obtain a *compact model* for the underlying data vector. 8

1.6 While the number of parameters needed to represent a large-scale data vector by a low-rank tensor model decreases exponentially in the order of the representation, it increases only linearly in the number of rank-1 terms. By leveraging this *blessing of dimensionality* in this thesis, we can tackle large-scale problems. 8

1.7 A low-rank approximation of a tensorized smooth function often provides an accurate representation. While only a few rank-1 terms already provide a good approximation, the accuracy can be improved by increasing the rank of the model. . . 9

1.8 By representing the solution of a linear system of equations by a low-rank tensor model in a similar fashion as before, we obtain an *implicit* tensor decomposition. In contrast to the decomposition of a tensor that is known explicitly, we want to compute a decomposition of a tensor that is only defined implicitly via the solution of a linear system. 11

1.9 While a tensor decomposition is a higher-order generalization of a matrix decomposition, a multilinear system of equations is a higher-order generalization of a linear system of equations. 11
1.10 Schematic overview of the thesis. 15

2.1 The multilinear SVD is a higher-order generalization of the well-known matrix SVD, enabling dimensionality reduction and subspace analysis for higher-order tensors. 23
2.2 The polyadic decomposition writes a tensor as a sum of rank-1 terms and is a fundamental tool for signal separation because it is unique under fairly mild conditions. 24
2.3 The more general block terms in a decomposition in multilinear rank-$(L_r, L_r, 1)$ terms allow us to model more complex phenomena than what is possible with the simple rank-1 terms of the canonical polyadic decomposition. 26

3.1 Tensor decompositions are a higher-order generalization of matrix decompositions and are well-known tools in many applications within various domains. Although multilinear systems are a generalization of linear systems in a similar way, this domain is relatively unexplored. LS-CPDs can be interpreted as multilinear systems of equations, providing a broad framework for the analysis of these types of problems. 34
3.2 Our algebraic method and optimization-based method (with random initialization) can perfectly reconstruct an exponential solution vector in the noiseless case. 47
3.3 Our optimization-based method (with random initialization) can reconstruct the rational solution vector in the noiseless case. 48
3.4 The naive method fails for an underdetermined LS-CPD while the NLS and algebraic method both perform well. The computational complexity of the algebraic method is much higher than the other two methods, especially for the highly underdetermined case (i.e., M close to the number of free variables). 49
3.5 By initializing the NLS algorithm with the algebraic solution instead of using a random initialization, fewer iterations are needed to achieve convergence. 50
3.6 Correct classification of a facial image under a *new* illumination condition. 53
3.7 The LS-CPD approach obtains more accurate results than the naive method and achieves similar accuracy as the dedicated OSACM method. The run-time of LS-CPD is slightly higher than OSACM for this example. 56

4.1 A low-rank model $\hat{\mathbf{S}}$ of the similarity measure provides a *good* approximation, allowing one to extract meaningful clusters using this approach [29]. 69

5.1 Blind Source Separation (BSS) can be reduced to the computation of a block term decomposition (BTD) by means of segmentation if the sources allow a low-rank representation, enabling a unique solution. 78

5.2 A low-rank representation of a reshaped *smooth* function often provides a good approximation. 81

5.3 By exploiting the intrinstic low-rank structure of the sources, the source signals can be perfectly reconstructed (in the noiseless case). 91

5.4 By exploiting the low-rank structure of the mixing coefficients *and* the sources, the mixing vectors and the source signals are perfectly reconstructed (in the noiseless case). 91

5.5 By exploiting the intrinstic low-rank structure of the sources, the source signals can even be perfectly recovered from *underdetermined* mixtures (in the noiseless case). 92

5.6 By increasing the number of samples, the relative error on the mixing vectors can be improved. This does not hold for the relative error on the sources because we also have to estimate longer source signals. Eventhough, the number of samples can be low in contrast to stochastic methods, the results are very accurate compared to the signal-to-noise ratio. 93

5.7 By using ill-conditioned mixing vectors, we simulate a *difficult* problem, hence, the computation is more difficult and the estimates are less accurate, as shown above. The effect is more pronounced for the source signals than the mixing vectors. . . 94

5.8 By computing the mixing vectors via the noisy observed data matrix and the pseudo-inverse of the estimated source matrix instead of extracting them directly from the decomposition, we can obtain a more accurate estimate of the mixing vectors when the rank-1 assumption does not hold entirely (i.e., for low α). Indeed, when the mixing vectors are less rank-1 like our model cannot attain a better estimate than the one given by Eckart–Young's theorem. 95

5.9 By increasing the rank of the model for the mixing vectors, one can decrease the relative error in the case of little rank-1-like mixing vectors (i.e., for low α). 95

5.10 What is considered a *good* choice of parameters will depend on the needs in a particular application: there is a clear trade-off between the compression rate and the accuracy. 96

5.11 By making a *good* choice of parameters, one can obtain an accurate low-rank model, even for real-life signals. 97

5.12 For increasing sample size, the relative errors on the estimates of the mixing matrix are getting more and more concentrated near zero. 98

5.13 By exploiting the intrinstic low-rank structure of the QRS complexes in the ECG, our method can achieve a clear separation of the fetal and maternal ECG. 99

5.14 Eventhough dedicated methods for direction-of-arrival estimation such as MUSIC and ESPRIT are more accurate than our method, we still obtain fairly accurate estimates using our general framework for both the line-of-sight and multipath scenarios. 101

5.15 Eventhough our segmentation-based method is less accurate than the dedicated MUSIC method, we simply have to increase the rank in order to cope with *near-field* sources. 102

5.16 In contrast to MUSIC, our method can distinguish direction-of-arrival angles that are close when only a few samples are available and the signal-to-noise ratio is low. 102

6.1 Decomposition of a fourth-order tensor $\mathcal{X}$ (illustrated as a third-order tensor) in (rank-P_r $\otimes$ vector) terms. One can see each factor matrix of the rank-P_r tensor as a petal of the flower and the vector as the stem, hence, “flower” decomposition. . . 110

6.2 Blind System Identification (BSI) can be reduced to the computation of a structured block term decomposition (BTD) by means of segmentation if the coefficients allow a low-rank representation, enabling a unique solution. 115

6.3 The transpose of a reshaped periodic signal is embedded in its ciruclant matrix. 118

6.4 A low-rank approximation of a reshaped smooth function often provides an accurate representation. Increasing the rank of the model improves the approximation. 119

6.5 By exploiting the low-rank structure of the FIR system coefficients, the coefficient vectors can be perfectly reconstructed (in the noiseless case). 128

6.6 The proposed method clearly obtains accurate results in comparison with the SNR for both the coefficients and the inputs, e.g., a relative error of -45 dB for 10 dB SNR. Increasing the number of samples improves the accuracy on the coefficients. This is not the case for the inputs due to the fact that one also has to estimate longer input signals. 129

6.7 Overestimating the exact system order $L = 2$ is not so critical for the accuracy on the coefficients. Underestimating the system order reduces the accuracy but the results are still quite good. 130
6.8 Although under- and overestimating the exact system order $L = 2$ decreases the accuracy on the inputs, the former provides slightly more accurate results than the latter. 130
6.9 Our segmentation-based approach allows a trade-off between compactness and accuracy of the model through the choice of the dimensions and the rank of the model. 131
6.10 By using a fat segmentation matrix, one can often attain a more accurate approximation than by using a tall segmentation matrix. 132
6.11 Although we overestimate the rank value of the zeroth coefficient vector of both inputs, the FIR system coefficients are perfectly reconstructed in the noiseless case. 133
6.12 The spectra of the obtained low-rank models $\mathbf{G}_1^{(0)}$ and $\mathbf{G}_2^{(0)}$ for the reshaped coefficient vectors of the zeroth delay clearly show a rank-1 model would have sufficed. 133
6.13 Illustration of a uniform rectangular array (URA) with $M = M_x M_y$ antennas: $M_x = 4$ and $M_y = 4$ in the x- and y-direction, respectively. The rth source is impinging on the URA from the far field and is characterized by two angles: the azimuth α_r and elevation β_r relative to the x-axis and the normal, respectively, hence, we have $-90^\circ \leq \alpha_r, \beta_r \leq 90^\circ$. . . 134
6.14 Segmentation is clearly more accurate than 2D-MUSIC. . . . 135
6.15 The loss in accuracy in a large-scale uniform rectangular array with missing antennas is limited, even for many inactive antennas *and* low SNR. 136
6.16 Simulated outputs of a high-density microelectrode array for measuring neuronal activity using simulated spike trains as inputs. 137
6.17 Our segmentation-based approach obtains an excellent separation of a convolutive mixture of simulated spike trains stemming from high-density microelectrode arrays for measuring neuronal activity. 138

7.1 Low-rank matrix or tensor models can often provide a parsimonious and accurate representation for smooth data. 140
7.2 By exploiting the rank-1 structure, the autoregressive coefficients are perfectly reconstructed (in the noiseless case). . . . 145
7.3 While our method obtains accurate results with respect to the signal-to-noise ratio, increasing the number of samples further improves the accuracy of the coefficient estimates. 146

7.4 While overestimation slightly reduces the accuracy, underestimating the system order fails to give accurate results. 146

8.1 Classification of a person that is included in the dataset. . . . 155
8.2 The MLSVD model captures the new person reasonably well. 157
8.3 Although we update the database with a new person using only one illumination condition, the KPE-based method recognizes that person in a new image under a different illumination condition. 157

9.1 Tensorization of an ECG data matrix into a third-order tensor with dimensions channel × time × heartbeat using segmentation. 163
9.2 Overall our method achieves good performance while better results can be obtained by using a suitable channel for a given subject, e.g., channels 8 and 6 achieve the highest specificity for subjects 1 and 4, respectively. 166
9.3 Fusing signals from multiple channels leads to a better performance. 166

List of tables

1.1 By representing a higher-order tensor by a compact tensor decomposition, we can alleviate or even break the curse of dimensionality. 7

3.1 The per-iteration computational complexity of the NLS algorithm for LS-CPD is dominated by the computation of the Jacobian. 45
3.2 Both PCs reduce the number of CG iterations in the underdetermined and square case. In the highly underdetermined case only the block-Jacobi PC can reduce the number of CG iterations. 51
3.3 The LS-CPD method for constructing a tensor with particular multilinear singular values is faster than APM. 54

4.1 By fully exploiting the Kronecker structure, we obtain a significant improvement in the computational complexity. 67
4.2 The block-Jacobi preconditioner (PC) effectively reduces the number of conjugate gradient (CG) iterations in various scenarios. 68

5.1 If the Hankelized version of an (exponential) polynomial has low rank, then the segmentized version has low rank as well, while the latter provides a more compact version than the former. 82
5.2 Segmentation enables possibly large compression ratios, indicating the applicability of this strategy for large-scale BSS. . 84

6.1 Dimensions of the obtained tensor $\mathcal{X}$ by applying segmentation to (6.8) using $M = 64$, $K = 1000$, and a given $\hat{N}$. 124
6.2 By exploiting more of the intrinsic higher-order structure in problem (6.8), more inputs can be identified. 124
6.3 By exploiting the block-Toeplitz structure in (6.8), more relaxed uniqueness conditions can be obtained. 127

8.1 By reformulating face recognition as a Kronecker product equation, higher performance (%) can be obtained in comparison to conventional techniques such as Eigenfaces as well as the tensor-based approach in [204]. 155
8.2 Higher performance (%) can be achieved by using multiple images under different illuminations. 156
8.3 When updating the database with a new person, our method can achieve higher accuracy (%) by fusing multiple images under different illumination conditions instead of using only one image of the new person. 157

A.1 Per-iteration computational complexity of the ingredients of the NLS algorithm. 185
A.2 Incorporating prior knowledge about the accuracy of the sensors in DOA estimation using weighted least squares allows one to decrease the relative error on the DOAs. 186

List of algorithms

3.1 Algebraic algorithm to solve a linear system of equations $\mathbf{Ax} = \mathbf{b}$ in which the solution $\mathbf{x}$ has a rank-1 CPD structure. 42

3.2 LS-CPD using Gauss–Newton with dogleg trust region. 44

4.1 Kronecker-structured linear system with a CPD constrained solution using Gauss–Newton with dogleg trust region. 64

6.1 Algebraic method for a decomposition of an $(N + 1)$th-order tensor $\mathcal{X}$ in (rank-$L_r \otimes$ vector) terms. 113

A.1 Weighted Least Squares with CPD-constrained weight tensor using Gauss–Newton and dogleg trust region 182

List of acronyms

ACMA	analytical constant modulus algorithm
ALS	alternating least squares
APM	alternating projection method
AR	autoregressive
ARX	autoregressive with exogenous terms
AVF	augmented vector foot
BCA	block component analysis
BPSK	binary phase shift keying
BSI	blind system identification
BSS	blind source separation
BTD	block term decomposition
CANDELINC	canonical decomposition with linear constraints
CG	conjugate gradients
cKPE	coupled Kronecker product equation
CM	constant modulus
CMA	constant modulus algorithm
COT	complex optimization toolbox
CPD	canonical polyadic decomposition
CS	compressed sensing
DOA	direction of arrival
ECG	electrocardiography
ECoG	electrocorticography
EEG	electroencephalography
ESPRIT	estimation of signal parameters via rotational invariance technique
EVD	eigenvalue decomposition
FECG	fetal ECG
FIR	finite impulse response
GEVD	generalized eigenvalue decomposition

GN	Gauss–Newton
HD-MEA	high-density micro-electrode array
HOS	higher-order statistics
HT	hierarchical Tucker
ICA	independent component analysis
KPE	Kronecker product equation
LMLRA	low multilinear rank approximation
LS	least squares
LS-CPD	linear system with CPD constrained solution
MECG	maternal ECG
MIMO	multiple input multiple output
MLSVD	multilinear singular value decomposition
MTKRPROD	matricized tensor Khatri–Rao product
MUSIC	multiple signal classification
NLS	nonlinear least squares
NMF	nonnegative matrix factorization
OSCMA	optimal step-size constant modulus algorithm
PC	preconditioner
PCA	principal component analysis
PCG	preconditioned conjugate gradients
PD	polyadic decomposition
PDE	partial differential equation
qN	quasi-Newton
SDF	structured data fusion
sEMG	surface electromyogram
SIMO	single input multiple output
SISO	single input single output
SNR	signal-to-noise ratio
SVD	singular value decomposition
TD	Tucker decomposition
TR	trust region
TT	tensor trains
ULA	uniform linear array
URA	uniform rectangular array
WBAN	wireless body area networks
WLS	weighted LS

List of symbols

Unary

$a, b, \ldots$	scalar
$\mathbf{a}, \mathbf{b}, \ldots$	vector
$\mathbf{A}, \mathbf{B}, \ldots$	matrix
$\mathcal{A}, \mathcal{B}, \ldots$	tensor
$\mathcal{I}, \mathcal{J}, \mathcal{K}$	index set
$.^{(n)}$	nth entry in (ordered) set
$\mathbf{T}_{(n)}$	mode-n unfolding of tensor $\mathcal{T}$
$\mathbb{R}$	real field
$\mathbb{C}$	complex field
$\mathbb{K}$	real or complex field
$\lvert \cdot \rvert$	absolute value or modulus
$\lvert\lvert\cdot\rvert\rvert$ or $\lvert\lvert\cdot\rvert\rvert_{\mathrm{F}}$	Frobenius norm
$.^{\mathrm{T}}$	transpose
$.^{\mathrm{H}}$	Hermitian transpose
$.^{-1}$	inverse
$.^{\dagger}$	Moore–Penrose pseudoinverse
$\bar{.}$	conjugation
$\mathcal{O}\left(\cdot\right)$	big-O notation
$\operatorname{vec}\left(\cdot\right)$	column-wise vectorization
$\operatorname{unvec}(\cdot)$	reshape vector into tensor
$\mathbf{I}$ or $\mathbf{I}_n$	identity matrix (of size $n \times n$)
$\mathbf{0}_n$	vector of zeros with length n
$\mathbf{0}_{m\times n}$	matrix of zeros with dimensions $m \times n$
$\mathbf{1}_n$	vector of ones with length n
$\mathbf{1}_{m\times n}$	matrix of ones with dimensions $m \times n$
$\lfloor x \rfloor$	floor (nearest integer smaller than or equal to)
$\lceil x \rceil$	ceil (nearest integer larger than or equal to)
$\mathcal{C}_n$	nth compound matrix
$\equiv$	equivalent
$\overset{\text{def}}{=}$	definition
d	derivative
∂	partial derivative

Binary

$\odot$	column-wise Khatri–Rao product
$\odot^{\mathrm{T}}$	row-wise Khatri–Rao product
$\otimes$	Kronecker product
$\mathbin{\scriptstyle\otimes}$	outer product
$\langle\cdot,\cdot\rangle$	inner product
$*$	Hadamard or element-wise product
$\cdot_n$	mode-n tensor matrix product

Other

$[\![\mathbf{A},\mathbf{B},\ldots]\!]$	polyadic decomposition with factors $\mathbf{A},\mathbf{B},\ldots$
$[\![\mathcal{G};\mathbf{A},\mathbf{B},\ldots]\!]$	low multilinear rank approximation with core tensor $\mathcal{G}$ and factors $\mathbf{A},\mathbf{B},\ldots$
$\mathrm{diag}(\mathbf{A})$	diagonal of matrix
$\mathrm{triu}(\mathbf{A})$	upper diagonal part of matrix
$\mathrm{diag}(\mathbf{a})$	diagonal matrix with entries in $\mathbf{a}$
$\mathrm{circ}(\mathbf{a})$	circulant matrix constructed with entries in $\mathbf{a}$
$\mathrm{gcd}(a)$	greatest common divisor of a
$\min(a,b,\ldots)$	smallest value of given set
$\mathrm{sign}(a)$	sign of a
$[\mathbf{a}\ \mathbf{b}\ \ldots]$	horizontal concatenation
$[\mathbf{a};\mathbf{b};\ldots]$	vertical concatenation
C_n^k	binomial coefficient

Part I

Introduction

Introduction

1

1.1 From two-way to multi-way data

Vectors and matrices are one-way and two-way arrays of numerical values, respectively, allowing us to represent various types of real-life data: a multi-sensor recording in a telecommunication application, a gray-scale image, or a movie rating table with user scores are all examples of data matrices. Matrix factorizations allow us to extract valuable information from the measured data matrix using the following simple but effective model:

$$\mathbf{X} = \mathbf{M} \cdot \mathbf{S}, \tag{1.1}$$

which has been employed successfully in various applications within signal processing, data mining, and machine learning. Examples are matrix-based compression for image, audio, or video data, matrix completion techniques to predict unknown values in recommender systems [33], and dimensionality reduction via principal component analysis (PCA) [121]. An important application in signal processing is blind source separation (BSS), which aims to recover the unknown mixing vectors in $\mathbf{M}$ and/or the unknown sources in $\mathbf{S}$ from the noisy data matrix $\mathbf{X}$ using the linear and instantaneous matrix model in (1.1) [42], [45], [50]. In order to interpret the mixing vectors and/or sources in real-life applications, it is important to obtain a *unique* solution. In general, however, a unique solution to (1.1) does not exist without imposing additional constraints because we can introduce any invertible matrix in such a way that we obtain an equally valid factorization of the given matrix:

$$\mathbf{X} = (\mathbf{M}\mathbf{W}) \cdot \left(\mathbf{W}^{-1}\mathbf{S}\right) = \tilde{\mathbf{M}} \cdot \tilde{\mathbf{S}}.$$

By imposing orthogonality, triangularity, or nonnegativity, one obtains the singular value decomposition (SVD), QR decomposition, or nonnegative matrix factorization (NMF), respectively. Although these constraints are useful in various applications, they are typically not suitable for BSS.

While matrix-based decompositions are powerful tools across a wide range

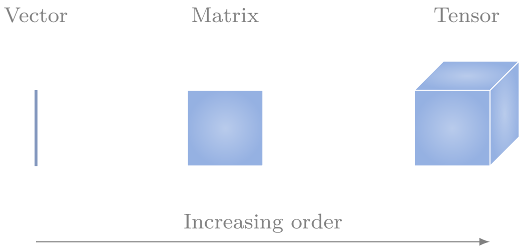


Figure 1.1: While a vector (first order) and a matrix (second order) represent one-way and two-way arrays of data, resp., a higher-order tensor represents a multi-way array of data.

of applications, matrices are limited to single-mode variations, making them less suited to tackle *multi-way* data. For example, multi-lead measurements of the electrocardiogram (ECG) signal of a single person can be stored in a matrix with modes channel × time, accommodating for the channel variation in the rows of the matrix. However, we cannot accommodate for different persons, rest states, and parameter settings in a straightforward way using only matrices. Higher-order tensors are a natural extension of vectors (first order) and matrices (second order), as shown in Figure 1.1, allowing us to represent multi-way data and therefore enabling us to tackle multiple modes of variation. Tensors are ubiquitous across various domains such as data mining, machine learning and signal processing [41], [125], [168]. Various techniques also employ a *tensorization* step to obtain a higher-order tensor and leverage the powerful utilities offered by higher-order tensors and tensor decompositions such as compression and uniqueness properties, among others [63]. For example, by segmenting and reshaping a multi-lead ECG signal, we obtain a tensor as shown in Figure 1.2.

Well-known matrix decompositions such as the SVD have been generalized to higher-order tensors, allowing us to use a *multilinear* SVD (MLSVD) [54]. The MLSVD, as shown in Figure 1.3, is closely related to the low-multilinear rank approximation (LMLRA) and the Tucker decomposition [125], [214], and has been successfully used in various applications for compression, dimensionality reduction, and subspace analysis [59], [98], [125], [214]. Another important tensor decomposition is the (canonical) polyadic decomposition (CPD) which decomposes a tensor into a (minimal) sum of rank-1 tensors. The decomposition is unique under fairly mild conditions [72], [73], [75], making the CPD a powerful tool in various signal processing applications. For example, a third-order tensor with one long mode is common in a signal processing context because that mode relates to the so-called “sample mode”. In that case, the CPD is unique with probability one if the number of rank-1 terms is bounded by the product of the other two tensor dimensions [47]. Various techniques (implicitly) reformulate the BSS problem in (1.1) into the computation of a CPD, enabling us to extract a unique solution from the

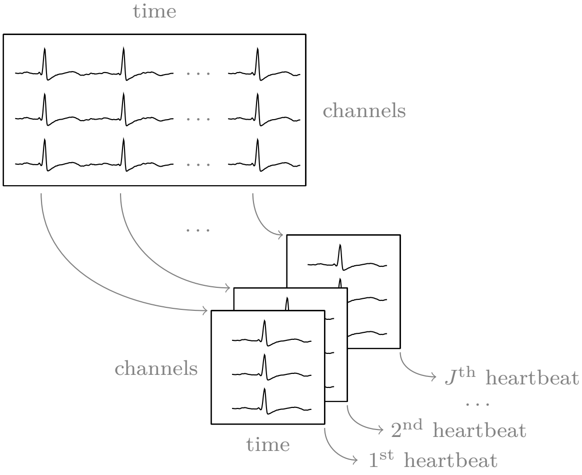

time
channels
...
channels
time
J^{th} heartbeat
...
2^{nd} heartbeat
1^{st} heartbeat

Figure 1.2: While tensors are by themselves already ubiquitous in a broad range of applications within signal processing, data mining, and machine learning, we also often want to construct a tensor from given data matrices. This is called *tensorization* and it allows us to leverage the powerful properties of higher-order tensors such as compression and uniqueness conditions. *In this example, we tensorize a given electrocardiogram data matrix by segmenting the heartbeats in each channel and then stacking them in a third-order tensor with modes channel × time × heartbeat. This figure is a reproduction of Figure 9.1; full details can be found in Chapter 9.*

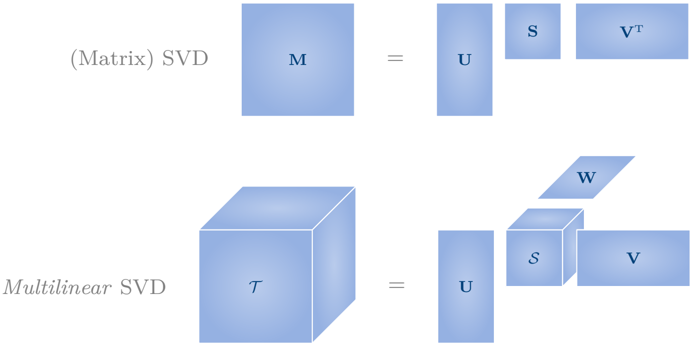


Figure 1.3: Matrix decompositions such as the well-known SVD can be generalized to the multilinear SVD, enabling compression and subspace analysis for higher-order tensors.

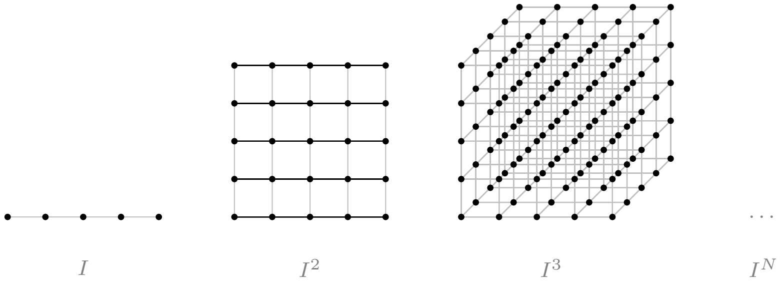


Figure 1.4: The number of entries in a higher-order tensor increases *exponentially* with the order of the tensor, which is known as the *curse of dimensionality*. *Given a vector, matrix, or third-order tensor with all dimensions equal to* I*, we have* I*,* I^2*, and* I^3 *entries, respectively, and, more generally,* I^N *entries for a* N*th-order tensor, which can explode,* even for small I*, when increasing the order* N*.*

decomposition under fairly mild conditions in contrast to the matrix case [1], [11], [45], [167]. Block term decompositions (BTD) allow us to model more complex phenomena than what is possible using the simple rank-1 terms of the CPD [48], [49], [51]–[53], [58]. We discuss tensor decompositions, uniqueness conditions, and computational methods in more detail in Chapter 2.

1.2 From a curse to a blessing

While the higher-order nature of tensors can accomodate for different modes of variation in data, the number of entries in a tensor increases exponentially with the tensor order as shown in Figure 1.4. This *curse of dimensionality* significantly limits the order of the tensors that can be handled in practical applications due to memory issues and computational requirements. Even for small dimensions, tensor problems are often large-scale; for example, a sixth-order tensor with size 100 for each mode, contains 10^{12} entries! The curse can, however, be alleviated or even broken by exploiting the *low-rank structure* by means of tensor decompositions as shown in Table 1.1. While we focus on a decomposition-based approach for full, but low-rank, tensors in this thesis, other techniques exist such as leveraging incomplete or sparse tensors and techniques employing approaches from compressed sensing (CS) [171], [208], [214].

While the MLSVD can be computed in a numerically stable way [54], [201], it can only alleviate the curse because we still need a core tensor of the same order as the original tensor. Although the core tensor can be much smaller than the original tensor, we still have an exponential dependence on the order which can blow up even for modest values. While the CPD allows us to effectively break the curse of dimensionality, it is a priori possible that

Table 1.1: By representing a higher-order tensor by a compact tensor decomposition, we can alleviate or even break the curse of dimensionality. *Given an* N*th-order tensor* $\mathcal{T}$ *with dimensions* $I_1 \times I_2 \times \cdots \times I_N$*, we assume that* $I_1 = I_2 = \cdots = I_N = I$*. For the low multilinear rank approximation, we assume that the multilinear rank equals* $(R, R, \ldots, R)$*. For the canonical polyadic decomposition,* R *equals the tensor rank. For the tensor train (TT) and hierarchical Tucker (HT) decomposition, we assume that the TT- and HT-ranks are equal to* R*.*

Decomposition	Number of variables
Low multilinear rank approximation	$\mathcal{O}\left(NIR + R^N\right)$
Canonical polyadic cecomposition	$\mathcal{O}\left(NIR\right)$
Tensor trains / matrix product states	$\mathcal{O}\left(2IR + (N-2)IR^2\right)$
Hierarchical Tucker decomposition	$\mathcal{O}\left((N-1)R^3 + NIR\right)$

degeneracy occurs [126], [172]. This means that, e.g., the magnitude of two terms grows without bounds but with opposite sign, resulting in numerical problems and a poor solution but a good fit. The problem can be avoided, however, by increasing the number of rank-1 terms or imposing constraints such as orthogonality or non-negativity [41], [126], [134], [185]. Tensor models such as tensor trains (TT) and hierarchical Tucker (HT) are often used in tensor-based scientific computing because they combine large compression rates with good numerical properties, allowing one to solve problems in a number of unknowns that exceeds the number of atoms in the universe [95], [96], [152].

While a pessimist would interpret the higher-order nature of tensors as a curse, an optimist sees an opportunity: by representing a large-scale vector by means of a low-rank tensor model, we can obtain a compact model for the large vector, as shown in Figure 1.5. Moreover, by representing the underlying vector by tensors of increasing order, we can obtain even more compact models. As illustrated in Figure 1.6, the number of parameters needed to represent the underlying vector decreases exponentially in the order of the tensor representation. This is a well-known strategy in tensor-based scientific computing and is also known as quantization [96], [99], [123]. The values for the dimensions of each mode of the tensor depend on the needs in a particular application and allow for a trade-off between accuracy and compression rate, as we will explain in Chapter 5 and Chapter 6.

The key assumption to employ this *blessing of dimensionality* is that the tensor representation has low rank such that we need only a few terms to accurately represent the underlying vector. Fortunately, this condition is often satisfied for large-scale problems [34] and for various signal models such as (exponential) polynomials, rational functions, and, in a broader sense, smooth signals, as shown in Figure 1.7 [51], [65], [95], [96], [123]. By leveraging this blessing of dimensionality in a novel way for instantaneous and convolutive BSS, we can tackle large-scale problems.

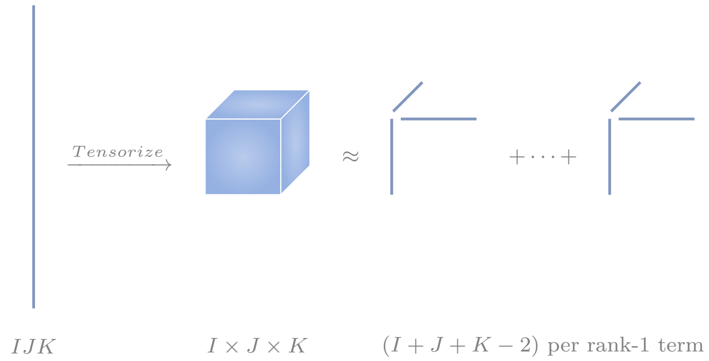
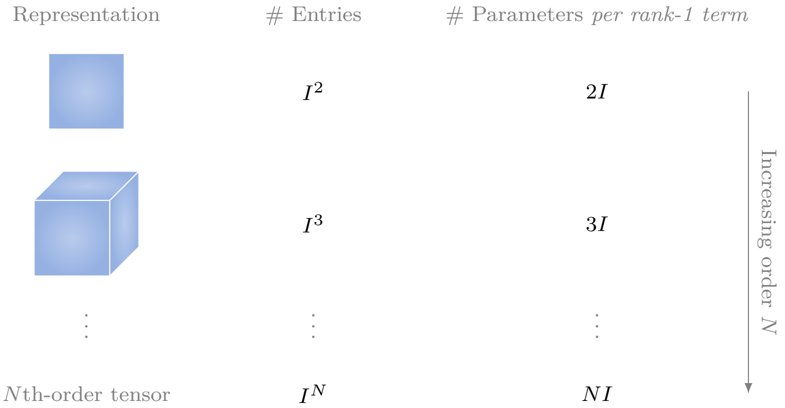


Figure 1.5: By approximating a tensorized representation of a large-scale data vector by a low-rank tensor decomposition, we obtain a *compact model* for the underlying data vector. *After reshaping a vector of length IJK into a third-order tensor of size $I \times J \times K$, we employ a canonical polyadic decomposition to represent the underlying vector with only $(I+J+K-2)$ parameters per rank-1 term; we need only $(I+J+K-2)$ instead of $(I+J+K)$ parameters because we compensate for scaling indeterminacies within a term.*



Representation	# Entries	# Parameters *per rank-1 term*
	I^2	$2I$
	I^3	$3I$
⋮	⋮	⋮
Nth-order tensor	I^N	NI

Figure 1.6: While the number of parameters needed to represent a large-scale data vector by a low-rank tensor model decreases exponentially in the order of the representation, it increases only linearly in the number of rank-1 terms. By leveraging this *blessing of dimensionality* in this thesis, we can tackle large-scale problems. *For each representation of dimensions $I_1 \times I_2 \times \cdots I_N$, we assume that $I_n = I$ for all n.*

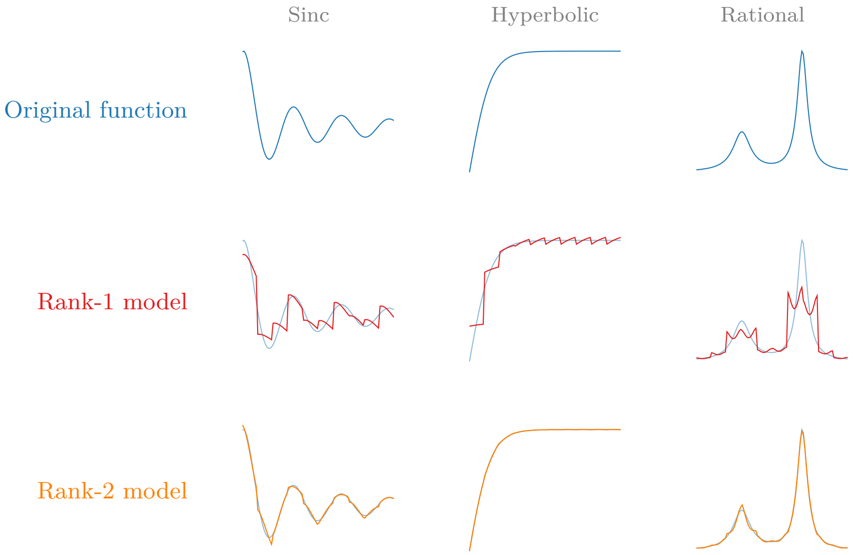


Figure 1.7: A low-rank approximation of a tensorized smooth function often provides an accurate representation. While only a few rank-1 terms already provide a good approximation, the accuracy can be improved by increasing the rank of the model. *This figure is a reproduction of Figure 6.4; full details can be found in Chapter 6.*

1.3 From explicit to implicit decompositions

Although the decomposition of a tensor that is known *explicitly* is a prevalent problem in signal processing and machine learning [41], [168], we often want to compute a decomposition of a tensor that is only known *implicitly*. Examples can be found in a wide range of domains such as signal processing [26], [221], system identification [17], [143], pattern recognition [23], [25], [26], [97], [197], [202], and scientific computing [8], [14], [94]–[96].

In this thesis, we focus on a particular type of implicit decomposition in the sense that we want to compute a decomposition of a tensor that is only known implicitly via the solution of a linear system of equations, as shown in Figure 1.8. We aim to develop a framework for implicit tensor decompositions in Chapter 3 and Chapter 4, enabling us to tackle various (large-scale) applications within signal processing, system identification, and pattern recognition. By *directly* solving the problem via optimization-based and algebraic methods, instead of first solving the unstructured system and then decomposing the reshaped solution, we can work much more efficiently and avoid error accumulation. By leveraging the compact representation of the solution vector in this way, we can tackle large-scale problems without constructing the tensor explicitly.

While leveraging the blessing of dimensionality for instantaneous and convolutive source separation gives rise to *explicit* tensor decomposition-based algorithms in Chapter 5 and Chapter 6, applying the same strategy to autoregressive (AR) system identification leads to *implicit* tensor decomposition-based algorithms in Chapter 7. The fairly mild uniqueness conditions of the CPD enable unique reconstruction and identification in large-scale source separation and system identification problems, respectively. Additionally, linear systems with a CPD constrained solution can be seen as multilinear systems, see Figure 1.9, which are a higher-order extension of linear systems of equations. By leveraging this interpretation, we develop a multilinear generalization of matrix-based classification, enabling tensor-based classification in Chapter 8 and Chapter 9.

1.4 Research objectives

In this thesis, we aim to develop *explicit* and *implicit* tensor decomposition-based algorithms with applications in blind source separation, (blind) system identification, and pattern recognition. We provide three main objectives.

Objective I: To provide *implicit* tensor decomposition-based algorithms.

While many algorithms have been proposed for the decomposition of a tensor that is known explicitly, implicit tensor decomposition-based algorithms have been less explored. Our objective is to develop state-of-the-art algebraic and

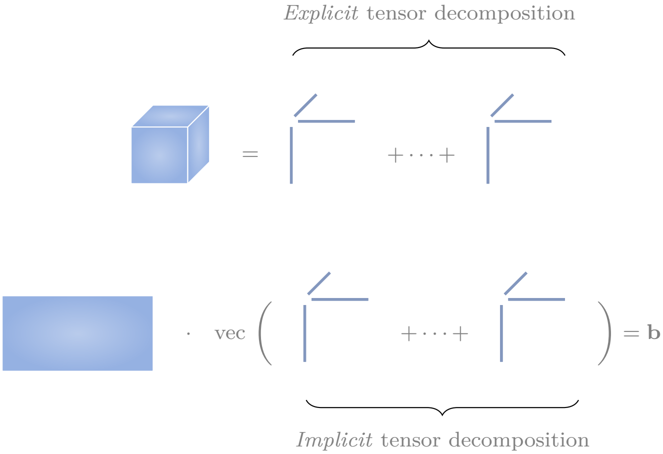

Explicit tensor decomposition
=
+ ··· +
vec
+ ··· +
= **b**
Implicit tensor decomposition

Figure 1.8: By representing the solution of a linear system of equations by a low-rank tensor model in a similar fashion as before, we obtain an *implicit* tensor decomposition. In contrast to the decomposition of a tensor that is known explicitly, we want to compute a decomposition of a tensor that is only defined implicitly via the solution of a linear system.

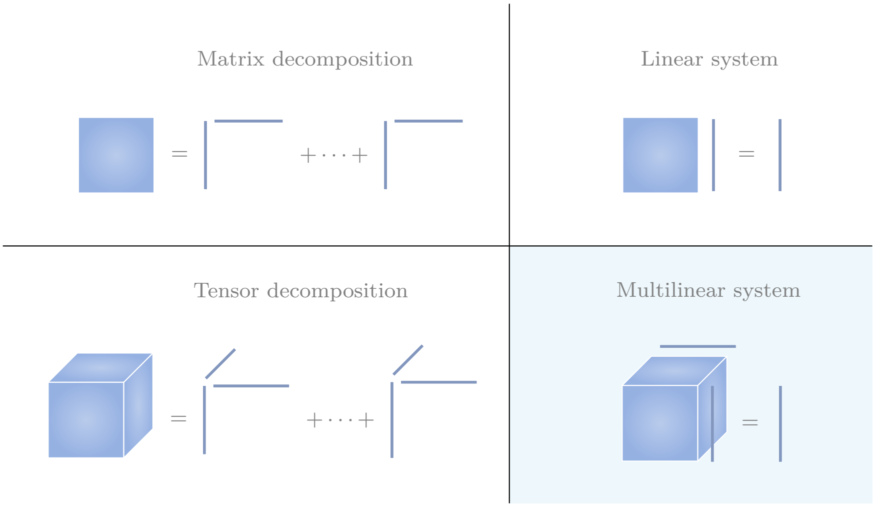


Figure 1.9: While a tensor decomposition is a higher-order generalization of a matrix decomposition, a multilinear system of equations is a higher-order generalization of a linear system of equations. *This figure is a reproduction of Figure 3.1; full details can be found in Chapter 3.*

optimization-based algorithms for the computation of a CPD that is known implicitly via the solution of a linear system of equations. This type of problem appears in a wide range of applications, but the CPD structure is often not recognized or not fully exploited.

Objective II: To leverage the *blessing of dimensionality* in large-scale (convolutive) signal separation and (blind) system identification.

Leveraging the blessing of dimensionality is a well-known strategy in tensor-based scientific computing to tackle large-scale problems. Our aim is to introduce this strategy in large-scale instantaneous and convolutive blind signal separation and system identification, yielding explicit and implicit tensor decomposition-based algorithms depending on the type of model.

Objective III: To develop a novel tensor-based classification scheme.

Linear systems with a CPD-constrained solution can be seen as multilinear systems, which are a generalization of linear systems of equations. By leveraging this interpretation, we aim to develop a novel tensor-based classification scheme with broad applicability.

The thesis contains three major parts:

- *Introduction.* The first part consists of this introductory chapter and an overview of *explicit* tensor decompositions. We discuss the state-of-the-art all-at-once optimization-based approach of [180], [210], [215] that will be used for the algorithmic development in the next part.

- *Algorithms.* In the second part, we propose a generic framework for *implicit* tensor decomposition-based algorithms. In Chapter 3, we develop algebraic and all-at-once optimization-based algorithms for a CPD that is known implicitly as the solution of a linear system of equations. By exploiting additional structure of the linear system, the computational complexity can be significantly reduced as we show for Kronecker-structured LS-CPD problems in Chapter 4.

- *Applications.* The third part comprises a set of applications within signal processing, system identification, and pattern recognition. By leveraging the blessing of dimensionality in large-scale instantaneous BSS, we obtain an *explicit* tensor decomposition-based algorithm in Chapter 5, enabling a *unique* solution and a way to cope with large-scale instantaneous BSS problems. In Chapter 6, we generalize the approach to convolutive mixtures, enabling large-scale *blind* system identification. By applying a similar approach for large-scale AR system identification, we obtain an *implicit* tensor decomposition-based algorithm in Chapter 7. Finally, we employ implicit tensor decompositions for tensor-based classification tasks such as face recognition and irregular heartbeat classification in Chapter 8 and Chapter 9, resp.

Each chapter of Part II and Part III is an adapted version of a paper that has been peer reviewed and contains research performed by the author of this thesis in collaboration with coauthors. Every chapter is *self-contained*, i.e., it contains a motivation, preliminaries, literature overview, methods, numerical experiments, and conclusion.

1.5 Thesis overview

We give a brief overview to all chapters below. The overall structure of the thesis is illustrated in Figure 1.10.

Part I: Introduction

In the remainder of this introduction, we give an overview of *explicit* tensor decompositions in Chapter 2. We focus on the CPD, MLSVD, and BTD because they are used extensively in the rest of the thesis. We discuss computational methods and give pointers to (generic) uniqueness conditions.

Part II: Algorithms

While Chapter 2 illustrates the importance of *explicit* tensor decomposition-based algorithms in signal processing and machine learning applications, we propose a general framework for *implicit* tensor decompositions in Chapter 3, allowing one to tackle various applications within signal processing, system identification, and pattern recognition. We limit ourselves to the computation of a CPD that is given implicitly as the solution of a linear system of equations (LS-CPD):

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{subject to} \quad \mathbf{x} = \operatorname{vec}(\text{CPD}). \tag{1.2}$$

The CPD structure significantly reduces the number of parameters needed to represent the solution of the linear system of equations, enabling one to solve (1.2) when the matrix $\mathbf{A}$ does not have full column rank, enabling a CS-style approach. In contrast to well-known CS reconstruction results that hold with a certain probability, we provide a uniqueness condition that holds *with probability one*: for random $\mathbf{A}$, we show that the CPD can be recovered uniquely, with probability one, if the number of equations is strictly larger than the number of free variables in the CPD. Equation (1.2) can also be interpreted as a *multilinear system of equations*, which is a generalization of linear systems of equations. In the linear case we have $\mathbf{A}\mathbf{x} = \mathbf{A} \cdot_2 \mathbf{x} = \mathbf{b}$, while in the multilinear case we have, e.g., $\mathcal{A} \cdot_2 \mathbf{u} \cdot_3 \mathbf{v} = \mathbf{b}$, which is the rank-1 case of (1.2). A naive method to solve (1.2) could be to first solve the linear system without structure and subsequently decompose a tensorized version unvec $(\mathbf{x})$ of the obtained solution. This approach works well if the linear system is overdetermined, but fails when $\mathbf{A}$ does not have full column rank. In contrast to the naive approach, we derive algebraic and *all-at-once* optimization-based methods that allow us to also solve the problem when the matrix $\mathbf{A}$ is fat or rank-deficient. Additionally, our approach allows us to work much more efficiently and avoid error accumulation. While we discuss LS-CPD applications extensively in Part III, we also illustrate the broad applicability of the proposed framework for three examples in this chapter: face recognition (see Chapter 8 for a detailed discussion), the construction of tensors with given multilinear singular values, and the deconvolution of constant modulus (CM) signals.

By fully exploiting the structure of $\mathbf{A}$, the computational complexity of the optimization-based algorithm can be significantly reduced, allowing us to tackle large-scale problems. In Chapter 4, we assume that $\mathbf{A}$ can be written as a sum of Kronecker products, which is a well-known strategy to reduce the computational complexity in various applications [9], [147], [198]. By leveraging the Kronecker structure, we show that the LS-CPD problem can be reformulated as a sum of CPDs with linearly constrained factor matrices. We derive an optimization-based algorithm that carefully exploits all available structure, enabling us to tackle large-scale problems. We illustrate our approach for graph clustering, demonstrating, once again, the wide applicability of our framework.

Part III: Applications

By leveraging the blessing of dimensionality for BSS in Chapter 5, we can uniquely recover the sources in large-scale problems. Our method models the sources by means of low-rank tensor models such as the CPD in order to obtain compact representations of the sources. After tensorizing the observed data matrix, we show that our method reduces to the computation of a tensor decomposition, allowing us to leverage mild uniqueness conditions. The deterministic tensorization technique employed in this step is called segmentation and is closely related to Hankel-based tensorization in the sense that segmentation is a compact version of Hankelization, allowing us to exactly represent (exponential) polynomials which can model a wide variety of signals. We show that the same strategy can be applied to the mixing coefficients of the BSS problem as in many large-scale applications the mixture is also compressible because of many and closely located sensors. Moreover, we combine both strategies, yielding a general *explicit* tensor decomposition-based technique that enables us to exploit the underlying structure of the sources and the mixtures *simultaneously*. We illustrate our method for fetal ECG extraction and direction-of-arrival (DOA) estimation in large-scale antenna arrays.

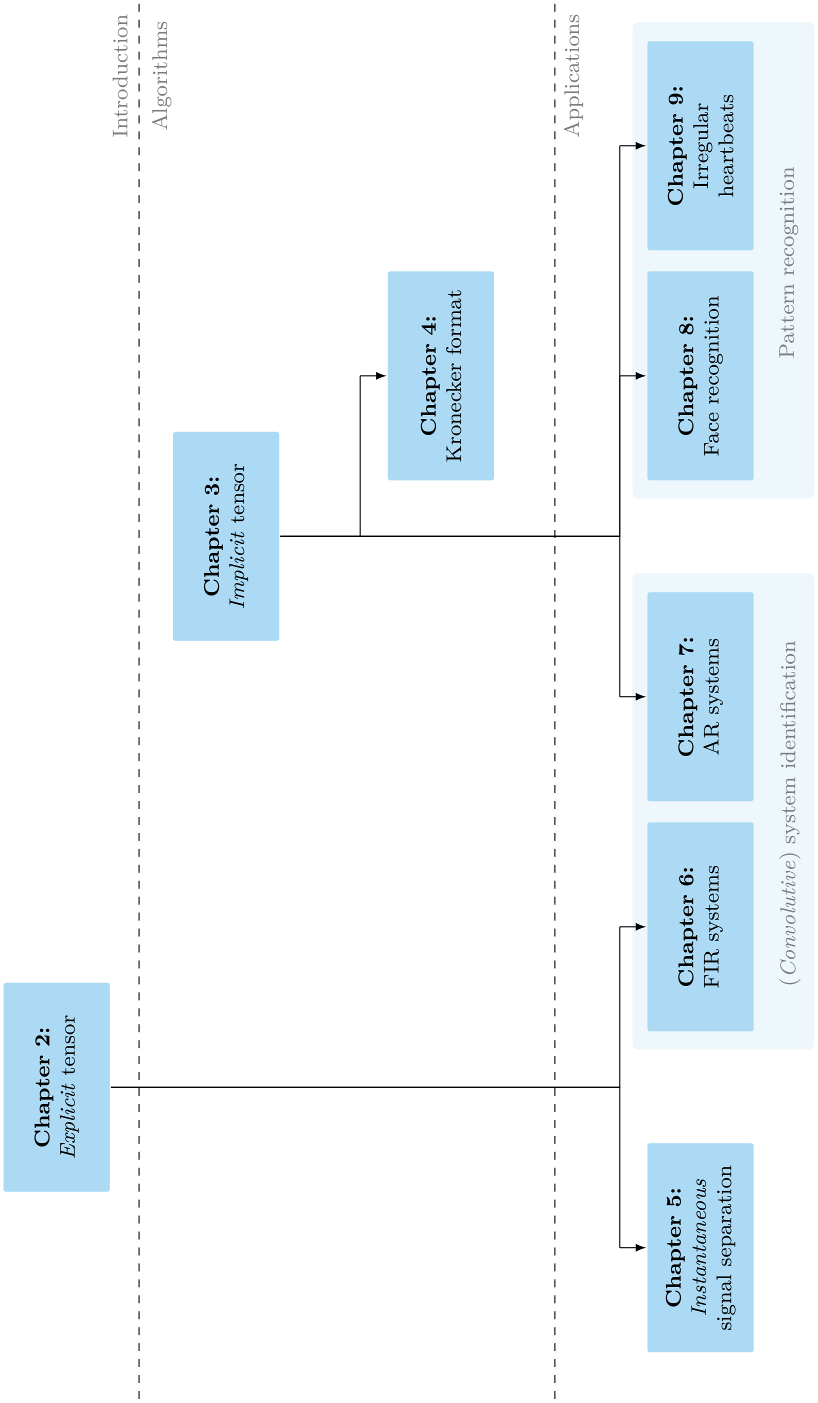

Introduction
Algorithms
Applications
Chapter 2: *Explicit* tensor
Chapter 3: *Implicit* tensor
Chapter 4: Kronecker format
Chapter 5: *Instantaneous* signal separation
Chapter 6: FIR systems
Chapter 7: AR systems
Chapter 8: Face recognition
Chapter 9: Irregular heartbeats
(*Convolutive*) system identification
Pattern recognition

Figure 1.10: Schematic overview of the thesis.

In Chapter 6, we generalize the segmentation-based approach for instantaneous BSS to the blind identification of finite impulse response (FIR) systems, allowing us to model large-scale *convolutive* mixtures of the inputs. We show that our method reduces the blind system identification (BSI) problem to a *structured* tensor decomposition of a tensor obtained by applying segmentation to the measured outputs, enabling a unique identification of the system and reconstruction of the inputs. We discuss the segmentation parameters in more detail and provide rules of thumb to find 'good' parameter values, enabling a trade-off between accuracy and compression rate. By leveraging the structure due to the convolutive nature of the FIR model, we can obtain more relaxed uniqueness results than results that do not take the structure into account. By exploiting the compact representation of low-rank tensor models, our method enables large-scale applications where conventional methods fall short. As such, we illustrate our method for DOA estimation in large-scale antenna arrays for uniform and non-uniform settings and scenarios with partially malfunctioning antennas. We also illustrate our method for neural spike sorting in high-density microelectrode arrays.

While the (blind) identification of FIR systems reduces to the computation of an *explicit* tensor decomposition, the identification of AR systems (using input *and* output data) reduces to the computation of an *implicit* tensor decomposition. By employing a similar strategy as before, we show in Chapter 7 that the identification of large-scale AR systems can be reformulated as the computation of an LS-CPD. The duality between explicit and implicit tensor decomposition-based algorithms emerges naturally for the identification of large-scale FIR and AR models, respectively, by means of segmentation.

By interpreting (1.2) as a multilinear system of equations, we can derive a multilinear generalization of matrix-based classification techniques such as EigenFaces [194]. For example, given the SVD of a labeled data matrix $\mathbf{M} = (\mathbf{US})\mathbf{V}^{\mathrm{T}}$, one can express a *new* data vector $\mathbf{m}^{(\text{new})}$ in the basis $(\mathbf{US})$ by solving a linear system of equations: $(\mathbf{US}) \cdot \mathbf{v}^{(\text{new})} = \mathbf{m}^{(\text{new})}$. By comparing the estimated coefficients with the rows of $\mathbf{V}$, one can classify the new data vector by assigning the label corresponding to the closest match. In a similar fashion, we propose a generic scheme for tensor-based classification by replacing the SVD by the *multilinear* SVD and the linear system by a *multilinear* system. Although we illustrate the technique for face recognition and irregular heartbeat classification in Chapter 8 and Chapter 9, respectively, it can be used for other classification tasks such as single-lead electroencephalography (EEG) classification [197] and atrial fibrillation detection [87].

Various parameters influence the performance of face recognition methods such as expression, pose, and illumination. In contrast to matrices, higher-order tensors enable us to naturally accommodate for the different modes of variation in real-life settings [203]. In Chapter 8, we show that tensor-based classification using the approach of TensorFaces [202] can be reformulated as the computation of an implicit tensor decomposition, yielding higher performance than matrix-based methods such as EigenFaces and other tensor-based techniques. More robust results can be achieved by using multiple images of the same person under different conditions, leading to *coupled* implicit decompositions. In order to add a new person to the database, existing methods require a facial image of the new person for each combination of condition, e.g., for each pose and illumination combination, which is a major drawback. We show that our method can add the new person to the database using only a few images with acceptable performance. We illustrate the performance of our method on the Extended Yale B dataset.

Cardiac arrhythmia are an important feature to assess the risk on sudden cardiac death. Automatic classification of irregular heartbeats is therefore an important part of ECG analysis. We propose a tensor-based method for single- and multi-lead heartbeat classification in Chapter 9 using a similar approach as in Chapter 8. Even though we use only one (or a few) leads instead of all leads to perform classification, we obtain similar performance as state-of-the-art matrix-based methods on the INCART dataset.

Conclusion and appendix

In Chapter 10, we summarize the main contributions of this thesis per chapter and provide directions for prospective work.

In Appendix A, we propose a *weighted* least-squares approach for the computation of a CPD of a higher-order tensor, enabling us to include prior knowledge about the noise in the least-squares cost function. Standard least-squares methods assume that the residuals are uncorrelated and have equal variances which is often not true in practice, rendering the approach suboptimal. We derive an optimization-based algorithm for the computation of a CPD using *low-rank* weights, enabling efficient weighting of the residuals. We illustrate our algorithm for DOA estimation using sensors with varying quality. In (near) real-time applications, it is possible to update the weights; see [199]. We also applied our approach to heartbeat morphology analysis [92]. We have included this contribution in the appendix because we want to emphasize the development of *implicit* tensor-decomposition based algorithms in the algorithmic part of this thesis.

Tensor decompositions

2

2.1 Introduction

While vectors and matrices represent one-way and two-way arrays of data, respectively, higher-order tensors represent multi-way arrays of data, allowing us to tackle multiple modes of variation. For example, a multi-channel ECG measurement of a single person can be represented by a matrix with modes channel and time, enabling us to extract valuable information through various well-known matrix-based tools. While multi-channel ECG measurements of multiple persons under different experimental conditions cannot be represented in a straightforward way using matrices, higher-order tensors can accommodate for the higher-order nature of the problem. By stacking the ECG measurements in a higher-order tensor with modes channel, time, persons, etc., we can employ powerful tensor-based techniques and potentially extract more information than what is possible using only matrix-based tools.

As a matter of fact, higher-order tensors are ubiquitous across a wide range of applications within domains such as data mining, machine learning, and (biomedical) signal processing [41], [112], [125], [155], [166]. Even though tensors arise naturally in many applications, various tensorization techniques exist that allow us to transform a known data matrix into a higher-order tensor, enabling us to leverage the powerful utitilies offered by tensor decomposition-based methods such as compression and mild uniqueness conditions [63].

In this chapter, we introduce basic concepts, definitions, and notations for higher-order tensors. Importantly, we discuss three (explicit) tensor decompositions that will be used extensively in this thesis: the multilinear singular value decomposition, the canonical polyadic decomposition, and a particular type of block term decomposition. We give pointers to applications, computational methods, and applications for all three decompositions. Finally, we discuss an *all-at-once* optimization-based framework to compute tensor decompositions that will be used for the development of algorithms in Part II.

2.2 Basic definitions and notations

2.2.1 Notations

We denote scalars, vectors, matrices, and tensors by lower case (e.g., a), bold lowercase (e.g., $\mathbf{a}$), bold uppercase (e.g., $\mathbf{A}$), and calligraphic letters (e.g., $\mathcal{A}$), respectively. The order of a tensor is equal to the number of modes, e.g., a tensor with three modes is a third-order tensor. The $(i_1, i_2, \ldots, i_N)$th entry of an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$, with $\mathbb{K}$ denoting either $\mathbb{R}$ or $\mathbb{C}$, is denoted by $a_{i_1 i_2 \ldots i_N}$. The nth element in a sequence is indicated by a superscript between parentheses, e.g., $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$. The transpose, Hermitian transpose, inverse, Moore–Penrose pseudoinverse, and conjugation are denoted by $\cdot^{\mathrm{T}}$, $\cdot^{\mathrm{H}}$, $\cdot^{-1}$, $\cdot^{\dagger}$, and $\bar{\cdot}$, respectively. The row-wise and column-wise concatenation of two vectors $\mathbf{a}$ and $\mathbf{b}$ is denoted by $\begin{bmatrix}\mathbf{a} & \mathbf{b}\end{bmatrix}$ and $\begin{bmatrix}\mathbf{a}; \mathbf{b}\end{bmatrix}$, respectively, with the latter being equal to $\begin{bmatrix}\mathbf{a}^{\mathrm{T}} & \mathbf{b}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$. A vector of ones or zeros of length N is denoted by $\mathbf{1}_N$ or $\mathbf{0}_N$, respectively. Similarly, we define a matrix of ones or zeros of size $M \times N$ as $\mathbf{1}_{M \times N}$ and $\mathbf{0}_{M \times N}$, respectively. Finally, we define the identity matrix of size $N \times N$ as $\mathbf{I}_N$.

2.2.2 Terminology

Definition 1. A *mode-n vector* of a tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ is a vector obtained by fixing every index except the nth and denoted as $\mathbf{t}_{i_1 \cdots i_{n-1} : i_{n+1} \cdots i_N}$.

A mode-n vector of a higher-order tensor is a natural extension of the columns (mode-1) and rows (mode-2) of a matrix.

Definition 2. A *nth-order slice* of an Nth-order tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ is a nth-order tensor obtained by fixing every index except n.

For example, the horizontal, frontal, and lateral (second-order) slices of a third-order tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$ are denoted by $\mathbf{X}_{i::}$, $\mathbf{X}_{:j:}$, and $\mathbf{X}_{::k}$, resp.

Definition 3. The *mode-n matrix representation* (or, alternatively, mode-n *unfolding*) of a higher-order tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ is a matrix $\mathbf{T}_{(n)} \in \mathbb{K}^{I_n \times I_1 I_2 \cdots I_{n-1} I_{n+1} \cdots I_{N-1} I_N}$ with the mode-n vectors of $\mathcal{T}$ as its columns.

Definition 4. *Vectorization* of a tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$, denoted as $\operatorname{vec}(\mathcal{T})$, maps each element $t_{i_1 i_2 \cdots i_N}$ onto $\operatorname{vec}(\mathcal{T})_j$ with $j = 1 + \sum_{k=1}^{N} (i_k - 1) J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$.

The operation $\operatorname{unvec}(\cdot)$ is the inverse of $\operatorname{vec}(\cdot)$.

2.2.3 Products

We introduce important matrix and tensor products and their relations.

Definition 5. Given two matrices $\mathbf{A} \in \mathbb{K}^{I \times J}$ and $\mathbf{B} \in \mathbb{K}^{K \times L}$, the *Kronecker product* $\mathbf{A} \otimes \mathbf{B} \in \mathbb{K}^{IK \times JL}$ is given by:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ldots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \cdots & a_{IJ}\mathbf{B} \end{bmatrix}.$$

Definition 6. Given two matrices $\mathbf{A} \in \mathbb{K}^{I \times K}$ and $\mathbf{B} \in \mathbb{K}^{J \times K}$, the (column-wise) *Khatri–Rao product* $\mathbf{A} \odot \mathbf{B} \in \mathbb{K}^{IJ \times K}$ is defined as the column-wise Kronecker product:

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} \mathbf{a}_1 \otimes \mathbf{b}_1 & \mathbf{a}_2 \otimes \mathbf{b}_2 & \cdots & \mathbf{a}_K \otimes \mathbf{b}_K \end{bmatrix}.$$

The *row-wise* Khatri–Rao product is denoted by $\odot^{\mathrm{T}}$ and is defined as the row-wise Kronecker product such that the following identity holds:

$$\mathbf{A} \odot^{\mathrm{T}} \mathbf{B} = \left(\mathbf{A}^{\mathrm{T}} \odot \mathbf{B}^{\mathrm{T}}\right)^{\mathrm{T}}.$$

Definition 7. Given two matrices $\mathbf{A} \in \mathbb{K}^{I \times J}$ and $\mathbf{B} \in \mathbb{K}^{I \times J}$, the *Hadamard product* (or element-wise product) $\mathbf{A} * \mathbf{B} \in \mathbb{K}^{I \times J}$ is given by:

$$\mathbf{A} * \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \cdots & a_{1J}b_{1J} \\ a_{21}b_{21} & a_{22}b_{22} & \cdots & a_{2J}b_{2J} \\ \vdots & \vdots & \ldots & \vdots \\ a_{I1}b_{I1} & a_{I2}b_{I2} & \cdots & a_{IJ}b_{IJ} \end{bmatrix}$$

Definition 8. Given two Nth-order tensors $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ and $\mathcal{B} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$, the *inner product* $\langle \mathcal{A}, \mathcal{B} \rangle$ is given by

$$\operatorname{vec}(\mathcal{B})^{\mathrm{H}} \operatorname{vec}(\mathcal{A}).$$

Definition 9. Given a Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ and and a Mth-order tensor $\mathcal{B} \in \mathbb{K}^{J_1 \times J_2 \times \cdots \times J_M}$, the *outer product* is a tensor $\mathcal{A} \circledcirc \mathcal{B} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N \times J_1 \times J_2 \times \cdots \times J_M}$ defined element-wise as:

$$\left(\mathcal{A} \circledcirc \mathcal{B}\right)_{i_1 i_2 \cdots i_N j_1 j_2 \cdots j_M} = a_{i_1 i_2 \cdots i_N} b_{j_1 j_2 \cdots j_M}.$$

For example, the outer product of two vectors $\mathbf{a} \in \mathbb{K}^{I}$ and $\mathbf{b} \in \mathbb{K}^{J}$ is a matrix $\mathbf{a} \circledcirc \mathbf{b} \in \mathbb{K}^{I \times J}$ of which the elements are defined as $\left(\mathbf{a} \circledcirc \mathbf{b}\right)_{ij} = a_i b_j$. The outer product is related to the Kronecker product via a vectorization:

$$\operatorname{vec}(\mathbf{a} \circledcirc \mathbf{b}) = \mathbf{b} \otimes \mathbf{a}.$$

Using the above definitions, one can obtain the following useful matrix identities [136], [212]:

$$\begin{aligned}
(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) &= (\mathbf{AC}) \otimes (\mathbf{BD})\,, \\
(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \odot \mathbf{D}) &= (\mathbf{AC}) \odot (\mathbf{BD})\,, \\
(\mathbf{A} \odot \mathbf{B})^{\mathrm{T}} (\mathbf{C} \odot \mathbf{D}) &= (\mathbf{A}^{\mathrm{T}}\mathbf{C}) * (\mathbf{B}^{\mathrm{T}}\mathbf{D})\,, \\
\operatorname{vec}(\mathbf{ABC}) &= (\mathbf{C}^{\mathrm{T}} \otimes \mathbf{A}) \operatorname{vec}(\mathbf{B})\,,
\end{aligned}$$

Definition 10. Given a tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $\mathbf{M} \in \mathbb{K}^{J_n \times I_n}$, the *mode-$n$ product* $\mathcal{T} \cdot_n \mathbf{M} \in \mathbb{K}^{I_1 \times \cdots I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N}$ is defined elementwise as:

$$(\mathcal{T} \cdot_n \mathbf{M})_{i_1 \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} t_{i_1 i_2 \cdots i_N} m_{j_n i_n}.$$

One can see that each mode-n vector of $\mathcal{T}$ is multiplied with the matrix $\mathbf{M}$, i.e., we have the following equivalent expression using the mode-n unfolding:

$$(\mathcal{T} \cdot_n \mathbf{M})_{(n)} = \mathbf{M}\mathbf{T}_{(n)}.$$

The mode-n product allows us to omit the use of (generalized) transposes in the higher-order case. For example, the matrix product $\mathbf{Y} = \mathbf{AXB}^{\mathrm{T}}$ is equivalent with $\mathbf{Y} = \mathbf{X} \cdot_1 \mathbf{A} \cdot_2 \mathbf{B}$, allowing one to generalize the equation in a straightforward way for higher order problems, e.g., $\mathcal{Y} = \mathcal{X} \cdot_1 \mathbf{A} \cdot_2 \mathbf{B} \cdot_3 \mathbf{C}$.

2.2.4 Rank

Definition 11. A *rank-1 tensor* of order N is a tensor that can be written as the outer product of N nonzero vectors.

Definition 12. The *rank* of a tensor $\mathcal{T}$ is equal to the minimal number of rank-1 terms that generate the tensor as their sum.

Definition 13. The *mode-n rank* of a Nth-order tensor is equal to the rank of the mode-n unfolding.

The mode-n rank is an extension of the column (mode-1) and row (mode-2) rank of matrix. Recall that the row rank, the column rank, and *the* rank are equal for matrices. For higher-order tensors, however, the mode-n ranks are not necessarily equal and the rank is not necessarily equal to any of the mode-n rank values [54].

Definition 14. The *multilinear rank* of a Nth-order tensor is equal to the N-tuple of mode-n rank values.

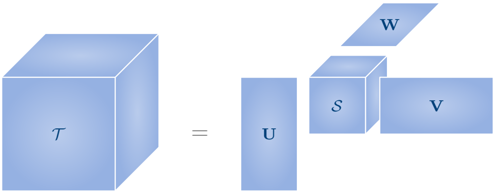


Figure 2.1: The multilinear SVD is a higher-order generalization of the well-known matrix SVD, enabling dimensionality reduction and subspace analysis for higher-order tensors.

2.2.5 Properties

Definition 15*.* The Nth-order tensors $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots I_N}$ and $\mathcal{B} \in \mathbb{K}^{I_1 \times I_2 \times \cdots I_N}$ are said to be *orthogonal* if the inner product $\langle \mathcal{A}, \mathcal{B} \rangle$ is equal to zero.

Definition 16*.* A Nth-order tensor $\mathcal{T}$ is said to be *all-orthogonal* if all slices of order $N-1$ are mutually orthogonal for each mode.

2.3 Tensor decompositions

2.3.1 Multilinear singular value decomposition (MLSVD)

The multilinear singular value decomposition (MLSVD) of a tensor is a higher-order extension of the matrix singular value decomposition (SVD) [41], [54], [168]. The decomposition describes a higher-order tensor by a set of factor matrices that are each related to a single mode and a core tensor that explains the interactions between the different modes, as shown in Figure 2.1.

Definition 17*.* The *multilinear singular value decomposition* (MLSVD) of a Nth-order tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ writes $\mathcal{T}$ as the product:

$$\mathcal{T} = \mathcal{S} \cdot_1 \mathbf{U}^{(1)} \cdot_2 \mathbf{U}^{(2)} \cdots \cdot_n \mathbf{U}^{(N)} \overset{\text{def}}{=} \left[\!\!\left[\mathcal{S}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)} \right]\!\!\right]$$

in which $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ is a unitary matrix, for $1 \leq n \leq N$, and the core tensor $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is ordered and all-orthogonal.

The core tensor is ordered in the sense that the slices have non-increasing Frobenius norm. The norms of the slices are equal to the mode-n singular values which are equal to the singular values of the mode-n unfoldings. The MLSVD has several other useful properties such as the fact that it is multilinear rank-revealing (in its number of significant multilinear singular values), has similar uniqueness properties as the matrix SVD, and can be used for subspace analysis, see [54] for a detailed discussion.

The MLSVD has been used successfully for dimensionality reduction, compression, and subspace analysis in various applications [41], [125], [168] within signal processing [59], [98], [146], [157], image processing [46], [147], [205],

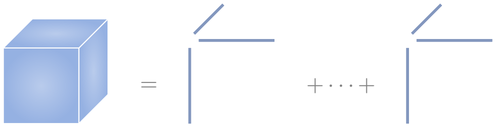


Figure 2.2: The polyadic decomposition writes a tensor as a sum of rank-1 terms and is a fundamental tool for signal separation because it is unique under fairly mild conditions.

pattern recognition [162], [203], [204], [216], and machine learning [168]. The MLSVD has also been used as a preprocessing step for the efficient computation of other tensor decompositions such as the canonical polyadic decomposition (CPD), see subsection 2.3.2 [28], [208], [210], [211], [213].

The MLSVD is closely related to the low-multilinear rank approximation (LMLRA) and the Tucker decomposition (TD); see, e.g., [54], [125], [214]. Several strategies to compute the decomposition have been proposed such as a SVD-based approach [54], [201], higher-order orthogonal iteration [56], and optimization-based methods [115], [116], [178], [180]. For large-scale tensors, efficient computational approaches have been proposed such as cross approximation [30], [140], [153] and a method using randomized SVDs [211].

2.3.2 Canonical polyadic decomposition (CPD)

The CPD is an important tool in many applications within signal processing and machine learning [41], [125], [168]. One of the reasons for the success of the CPD is the fact that the decomposition is unique under fairly mild uniqueness conditions [72], [75], yielding a powerful advantage over matrices in many applications such as blind source separation (BSS) and blind system identification (BSI) [168].

Definition 18. A *polyadic decomposition* (PD) writes a Nth-order tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ as a sum of R rank-1 terms:

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(N)} \overset{\text{def}}{=} \left[\!\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)} \right]\!\!\right]. \tag{2.1}$$

The columns of the factor matrices $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$ are equal to the factor vectors $\mathbf{u}_r^{(n)}$ for $1 \leq r \leq R$. The PD is called *canonical* (CPD) when R is equal to the rank of $\mathcal{T}$.

The CPD is a popular tensor decomposition in a wide range of applications within signal processing such as instantaneous and convolutive BSS [11], [45], [51], [57], [64], [187], sensor array processing [141], [167], [181]–[183], and telecommunications [53], [169], [170]. In biomedical applications [112], the decomposition has been employed for the analysis of ECG [92], [155], EEG [2], [10], [61], [66], [142], [145], [223] and fMRI data [15], [36], [113]. The CPD has also been used successfully in data mining and machine learning applications [5], [6], [12], [122], [168], [202].

The CPD is said to be *essentially unique* if the decomposition is unique up to trivial permutation of the rank-1 terms and scaling and counterscaling of the factors within the same rank-1 term. In general, no unique solution exists in the matrix case, i.e., for $N = 2$, without additional assumptions for $R > 1$. For tensors, i.e., for $N > 2$, we typically expect uniqueness under fairly mild conditions. Moreover, for increasing order N, we expect even milder uniqueness conditions [166], [184], [186]. *Deterministic* uniqueness conditions guarantee uniqueness for a particular choice of factor matrices satisfying the condition. For example, consider the following uniqueness result:

Theorem 1. Given a CPD $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ of a tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$ as in (2.1) with $N = 3$. If the matrices $\mathbf{A}$ and $\mathbf{B}$ have full column rank and $\mathbf{C}$ does not have proportional columns, the decomposition is essentially unique.

Generic uniqueness conditions consider uniqueness with probability one when the entries of the factor matrices are drawn from absolutely continuous probability density functions. As such, the generic counterpart of Theorem 1 implies uniqueness if $I \geq R$, $J \geq R$, and $K \geq 2$. In practice, however, *milder* uniqueness conditions than Theorem 1 can be obtained. For example, the CPD is generically unique (with a few known exceptions) if [39]:

$$R \leq \left\lceil \frac{IJK}{I + J + K - 2} \right\rceil - 1 \quad \text{and} \quad IJK \leq 15000,$$

with $\lceil x \rceil$ the smallest integer not less than x. The bound on the number of entries IJK has only been verified numerically up to 15000, but is assumed to hold for larger number of entries as well. State-of-the-art deterministic and generic uniqueness conditions for higher-order tensors can be found in [39], [70]–[75], [135], [184], [186] and references therein. CPD existence and uniqueness results in the *noisy* case are considered in [84]. For a brief introduction to CPD uniqueness we refer to [168, Section IV].

Many algorithms have been proposed for the computation of (coupled and constrained) CPDs such as algebraic methods [47], [72], [75], [181], [186], alternating least-squares (ALS) techniques [125], and *all-at-once* optimization-based algorithms [4], [154], [158], [178], [180], [189], [211]. For example, if a CPD of a third-order tensor satisfies Theorem 1, it can be found algebraically by means of a generalized eigenvalue decomposition (GEVD) [103]. While ALS methods are popular thanks to their straightforward implementation and speed for simple problems, they are often outperformed in ill-conditioned cases by optimization-based techniques such as qN and NLS algorithms [210]. Additionally, these sophisticated optimization techniques have favorable convergence properties and are more robust in practice [150], [180], [210]. We discuss the *all-at-once* optimization framework in more detail in section 2.4.

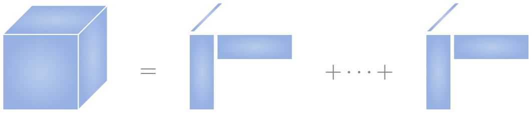


Figure 2.3: The more general block terms in a decomposition in multilinear rank-$(L_r, L_r, 1)$ terms allow us to model more complex phenomena than what is possible with the simple rank-1 terms of the canonical polyadic decomposition.

2.3.3 Block term decomposition (BTD)

BTDs are a generalization of the CPD and the MLSVD in the sense that they decompose a tensor into a sum of *blocks* with a particular multilinear rank, enabling us to model more complex phenomena [49]. While several variants exist, we focus on a BTD that decomposes a tensor in multilinear rank-$(L_r, L_r, 1)$ terms, as illustrated in Figure 2.3. Other BTDs, associated uniqueness results, and algorithms can be found in [48], [49], [51], [58].

Definition 19. A *block term decomposition* (BTD) writes a third-order tensor $\mathcal{X} \in \mathbb{K}^{I\times J\times K}$ as a sum of R *multilinear rank-*$(L_r, L_r, 1)$ *terms*:

$$\mathcal{X} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \otimes \mathbf{c}_r, \tag{2.2}$$

in which $\mathbf{A}_r \in \mathbb{K}^{I\times L_r}$ and $\mathbf{B}_r \in \mathbb{K}^{J\times L_r}$ have full column rank L_r and $\mathbf{c}_r$ is a nonzero vector.

The decomposition in (2.2) can be interpreted as a CPD with proportional columns in the third factor matrix. By defining the following factor matrices:

$$\begin{aligned}
\mathbf{A} &= \begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2 & \cdots & \mathbf{A}_R\end{bmatrix} \in \mathbb{K}^{I\times R'},\\
\mathbf{B} &= \begin{bmatrix}\mathbf{B}_1 & \mathbf{B}_2 & \cdots & \mathbf{B}_R\end{bmatrix} \in \mathbb{K}^{J\times R'},\\
\mathbf{C}^{(\mathrm{ext})} &= \begin{bmatrix}\mathbf{1}_{L_1}^{\mathrm{T}} \otimes \mathbf{c}_1 & \mathbf{1}_{L_2}^{\mathrm{T}} \otimes \mathbf{c}_2 & \cdots & \mathbf{1}_{L_R}^{\mathrm{T}} \otimes \mathbf{c}_R\end{bmatrix} \in \mathbb{K}^{K\times R'}
\end{aligned}$$

with $R' = \sum_{r=1}^{R} L_r$, we can reformulate (2.2) as a rank-R' CPD of $\mathcal{X}$:

$$\mathcal{X} = \left[\!\left[\mathbf{A}, \mathbf{B}, \mathbf{C}^{(\mathrm{ext})}\right]\!\right].$$

The block terms are more general than the simple rank-1 terms of a PD, enabling us to model more complex phenomena [52] in various applications such as BSS [51], [65], blind deconvolution [53], [187], biomedical signal processing [36], [111], [151], [218], and system identification [79]. The values of the parameters L_r can be estimated algebraically under some conditions [76].

The BTD in multilinear rank-$(L_r, L_r, 1)$ terms is *essentially unique* if it is unique up to trivial permutation of the rth and r'th term, if $L_r = L_{r'}$, and scaling and counter-scaling of $(\mathbf{A}_r\mathbf{B}_r^{\mathrm{T}})$ and $\mathbf{c}_r$ in the same term. For example, consider the following uniqueness condition; see [49] for a detailed discussion.

Theorem 2. Given a BTD in multilinear rank-$(L_r, L_r, 1)$ terms of a tensor $\mathcal{X} \in \mathbb{K}^{I\times J\times K}$ as in (2.2) with $I, J \geq R'$. If the matrices $\mathbf{A}$ and $\mathbf{B}$ have full column rank and $\mathbf{C} = \begin{bmatrix}\mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_R\end{bmatrix}$ does not have proportional columns, the decomposition is essentially unique.

Theorem 2 is the counterpart of Theorem 1 for CPDs.

2.4 *All-at-once* optimization-based algorithms

In Part II, we develop *all-at-once* optimization-based algorithms for *implicit* tensor decompositions using the complex optimization toolbox (COT) as a numerical optimization solver [177], [179]. This framework provides quasi-Newton (qN) and nonlinear-least squares (NLS) implementations as well as line search, plane search and trust-region methods [150]. We aim to develop algorithms in the same spirit as the optimization-based algorithms for *explicit* tensor decompositions in Tensorlab [178], [180], [211], [212], [215]. While we provide a brief overview of the optimization framework in this section, we refer the interested reader to a more detailed discussion in [210].

In this thesis, we aim to develop optimization-based algorithms using a NLS approach. The goal of NLS is to minimize the squared error between a known data vector $\mathbf{t}$ and a nonlinear model $m(\mathbf{z})$ with optimization variables $\mathbf{z}$. Mathematically, we obtain the following optimization problem:

$$\min_{\mathbf{z}} f(\mathbf{z}) \quad \text{with} \quad f(\mathbf{z}) = ||m(\mathbf{z}) - \mathbf{t}||_{\mathrm{F}}^2 \tag{2.3}$$

using the Frobenius norm. For example, the computation of a CPD of a known third-order tensor $\mathcal{T}$ can be formulated as problem (2.3) by taking $\mathbf{t} = \operatorname{vec}(\mathcal{T})$, $\mathbf{z} = \begin{bmatrix}\operatorname{vec}(\mathbf{A}); \operatorname{vec}(\mathbf{B}); \operatorname{vec}(\mathbf{C})\end{bmatrix}$, and model $m(\mathbf{z}) = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$.

While we focus on the implementation of GN-style methods, the expressions can be used for other qN and NLS algorithms as well. A GN method using a trust region (TR) approach solves the optimization problem in (2.3) by linearizing the residual $\mathbf{r} = m(\mathbf{z}) - \mathbf{t}$ in each iteration k and subsequently solving the following least-squares (LS) problem [150], [210]:

$$\min_{\mathbf{p}_k} \frac{1}{2} ||\mathbf{r}_k + \mathbf{J}_k\mathbf{p}_k||_{\mathrm{F}}^2 \quad \text{subject to} \quad ||\mathbf{p}_k|| \leq \Delta_k \tag{2.4}$$

with step $\mathbf{p}_k = \mathbf{z}_{k+1} - \mathbf{z}_k$, Jacobian $\mathbf{J} = \mathrm{d}\mathbf{r}/\mathrm{d}\mathbf{z}$, and trust-region Δ_k. The exact solution to (2.4) is given by the following linear system [150]:

$$\mathbf{H}_k\mathbf{p}_k = -\overline{\mathbf{g}}_k \tag{2.5}$$

with $\mathbf{H}$ being the Hessian and the conjugated gradient $\overline{\mathbf{g}} = (\partial f/\partial \mathbf{z})^{\mathrm{H}}$. The

Hessian is typically approximated in some way because it is often expensive or difficult to compute explicitly [210]. In the GN method we approximate the Hessian with the Gramian of the Jacobian in each iteration, i.e., we use:

$$\mathbf{H}_k = \mathbf{J}_k^{\mathrm{H}} \mathbf{J}_k.$$

The variables can then be updated *all at once* in each iteration as:

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \mathbf{p}_k.$$

The linear system in (2.5) can be solved exactly for small-scale problems, but for large-scale problems we typically use several preconditioned conjugated gradient (PCG) iterations in order to reduce the computational complexity.

Part II

Algorithms

Linear systems with a canonical polyadic decomposition constrained solution: algorithms and applications

3

ABSTRACT | Real-life data often exhibit some structure and/or sparsity, allowing one to use parsimonious models for compact representation and approximation. When considering matrix and tensor data, low-rank models such as the (multilinear) singular value decomposition (SVD), canonical polyadic decomposition (CPD), tensor train (TT), and hierachical Tucker (HT) model are very common. The solution of (large-scale) linear systems is often structured in a similar way, allowing one to use compact matrix and tensor models as well. In this chapter we focus on linear systems with a CPD constrained solution (LS-CPD). Our main contribution is the development of optimization-based and algebraic methods to solve LS-CPDs. Furthermore, we propose a condition that guarantees generic uniqueness of the obtained solution. We also show that LS-CPDs provide a broad framework for the analysis of multilinear systems of equations. The latter are a higher-order generalization of linear systems, similar to tensor decompositions being a generalization of matrix decompositions. The wide applicability of LS-CPDs in domains such as classification, multilinear algebra, and signal processing is illustrated.

This chapter is a slightly adapted version of M. Boussé, N. Vervliet, I. Domanov, O. Debals, and L. De Lathauwer, “Linear systems with a canonical polyadic decomposition constrained solution: Algorithms and applications”, *Numerical Linear Algebra with Applications*, vol. 25, no. 6, e2190, Aug. 2018. The figures have been updated for consistency.

3.1 Introduction

Real-life data can often be modeled using compact representations because of some intrinsic structure and/or sparsity [34]. Well-known representations are low-rank matrix and tensor models such as nonnegative matrix factorization (NMF), the (multilinear) singular value decomposition (SVD), canonical polyadic decomposition (CPD), tensor train (TT), and hierarchical Tucker (HT) models [41], [54], [90], [95], [125], [152], [168]. Examples of data that can be represented or well approximated by such models are exponential polynomials, rational functions (and in a broader sense smooth signals), as well as periodic functions [20], [21], [96], [123]. When dealing with vector/matrix data, one often reshapes the data into higher-order tensors which are then modeled using low-rank approximations, enabling efficient processing in the compressed format. This strategy has been used in tensor-based scientific computing and signal processing to handle various large-scale problems [20], [21], [96], [99].

Similarly, the solution of a (large-scale) linear system can often be expressed by a low-rank tensor. Such problems are well-known in tensor-based scientific computing; see [96]. They arise, e.g., after discretizing high-dimensional partial differential equations (PDEs). The low-rank model ensures efficient computations and a compact representation of the solution. In such large-scale problems, one often assumes that the coefficient matrix and/or right-hand side have some additional structure or can also be expressed using a tensor model. Several methods have been developed for linear systems with a Kronecker-structured coefficient matrix and a CPD structured solution such as the projection method [8], alternating least squares (ALS) [14], and a gradient method [83]. TTs or HT models are also often used because they combine large compression rates and good numerical properties [95], [152].

In this chapter, we present a new framework for linear systems of equations with a CPD constrained solution, abbreviated as LS-CPD. In other words, we want to solve linear systems of the form:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{with} \quad \mathbf{x} = \operatorname{vec}\left(\text{CPD}\right),$$

in which $\operatorname{vec}(\cdot)$ is a vectorization. A simple second-order rank-1 example is $\mathbf{x} = \operatorname{vec}(\mathbf{u} \otimes \mathbf{v})$ with $\otimes$ the outer product. In particular we develop algebraic as well as optimization-based algorithms that properly address the CPD structure. A naive method to solve LS-CPDs could be to first solve $\mathbf{A}\mathbf{x} = \mathbf{b}$ without structure and subsequently decompose a tensorized version $\operatorname{unvec}(\mathbf{x})$ of the obtained solution. This approach works well if the linear system is overdetermined, but, in contrast to our algebraic and optimization-based methods, fails in the underdetermined case. The proposed optimization-based method computes a solution of the LS-CPD problem by minimizing a least-squares objective function. We have derived expressions for the gradient, Jacobian, and approximation of the Hessian which are the ingredients for standard quasi-Newton (qN) and nonlinear least squares (NLS) techniques. We use the complex optimization framework in Tensorlab, a toolbox for tensor computations in MATLAB, as a numerical optimization solver [177], [179], [180], [215]. The optimization-based methods allow us to work much more efficiently and avoid error accumulation in contrast to the naive or algebraic methods. The latter two methods can be used to obtain a good initialization for the optimization-based methods when considering perturbed LS-CPD problems. Our framework can be extended to other tensor decompositions such as the block term decomposition (BTD), multilinear SVD (MLSVD), low-multilinear rank approximation (LMLRA), TTs or HT models [49], [54], [96], [168].

Furthermore, LS-CPDs can be interpreted as multilinear systems of equations which are a generalization of linear systems of equations. The latter can be expressed by a matrix-vector product between the coefficient matrix and the solution vector, e.g., $\mathbf{A}\mathbf{x} = \mathbf{b}$, or, equivalently, $\mathbf{A} \cdot_2 \mathbf{x}^{\mathrm{T}}$, using the mode-$n$ product [125]. The generalization to a multilinear system is then straightforward because it can be expressed by tensor-vector products between the coefficient tensor and multiple solution vectors: $\mathcal{A} \cdot_2 \mathbf{x}^{\mathrm{T}} \cdot_3 \mathbf{y}^{\mathrm{T}} = \mathbf{b}$. This is very similar to tensor decompositions which are higher-order generalizations of matrix decompositions [41], [125], [168]. However, in contrast to tensor decompositions, the domain of multilinear systems is relatively unexplored. To the best of the authors' knowledge, only a few cases have been studied in a disparate manner such as the fully symmetric rank-1 tensor case with a particular coefficient structure [68], sets of bilinear equations [7], [44], [120], and a particular type of multilinear systems that can be solved via so-called tensor inversion [27]. LS-CPDs provide a general framework to solve multilinear systems; see Figure 3.1.

The CPD structure in LS-CPDs strongly reduces the number of parameters needed to represent the solution. For example, a cubic third-order tensor of size $I \times I \times I$ contains I^3 entries but its CPD needs only $\mathcal{O}(3RI)$ parameters with R being the number of terms in the decomposition. The possibly very compact representation of the solution enables one to solve the LS-CPD problem for the underdetermined case in a compressed-sensing (CS) style [34], [78]. A similar idea has been studied for the low-rank matrix case [193]. In contrast to well-known CS reconstruction conditions, we derive a uniqueness condition for LS-CPDs that holds with probability one. In particular, we derive a generic uniqueness condition for the solution $\mathbf{x}$ of the LS-CPD problem given a coefficient matrix $\mathbf{A}$ of which the entries are drawn from absolutely continuous probability density functions.

LS-CPDs appear in a wide range of applications; see, e.g., [149], [188], [204], but the CPD structure is often not recognized or not fully exploited. In this chapter, the applicability of LS-CPDs is illustrated in three different

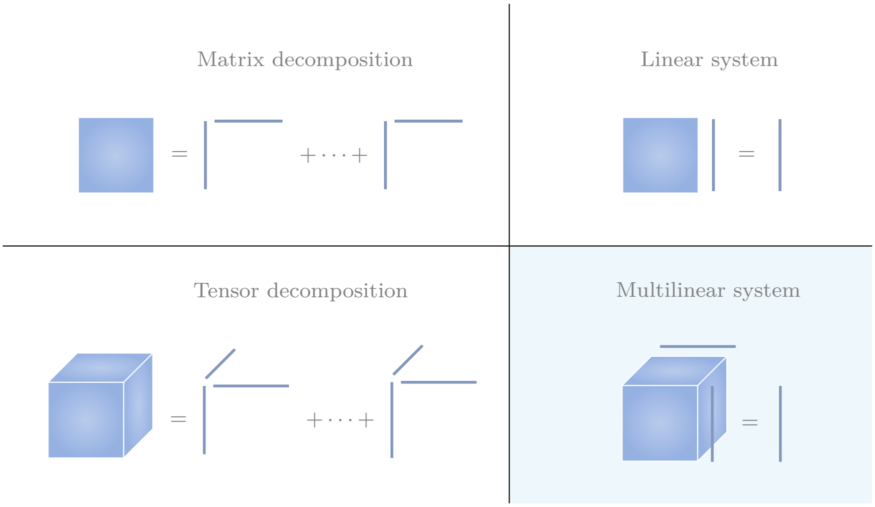


Figure 3.1: Tensor decompositions are a higher-order generalization of matrix decompositions and are well-known tools in many applications within various domains. Although multilinear systems are a generalization of linear systems in a similar way, this domain is relatively unexplored. LS-CPDs can be interpreted as multilinear systems of equations, providing a broad framework for the analysis of these types of problems.

domains: classification, multilinear algebra, and signal processing. In the first case, we show that tensor-based classification can be formulated as the computation of an LS-CPD. Although we illustrate the technique with face recognition [25], one can consider other classification tasks such as irregular heartbeat classification and various computer vision problems [23], [202]–[204]. Next, the construction of a real-valued tensor that has particular multilinear singular values is formulated as an LS-CPD. By properly exploiting the symmetry in the resulting problem, our method is faster than literature methods. We conclude with the blind deconvolution of constant modulus (CM) signals such as 4-QAM or BPSK signals using LS-CPDs.

In the remainder of this introduction, we give an overview of the notation, basic definitions, and multilinear algebra prerequisites. In section 3.2, we define LS-CPDs and briefly discuss structure and generic uniqueness. Next, we develop an algebraic algorithm and an optimization-based algorithm to compute LS-CPDs in section 3.3. Numerical experiments and applications are presented in sections 3.4 and 3.5, respectively. We conclude the chapter and discuss possible future work in section 3.6.

3.1.1 Notation and definitions

A tensor is a higher-order generalization of a vector (first-order) and a matrix (second-order). We denote tensors by calligraphic letters, e.g., $\mathcal{A}$. Vectors and matrices are denoted by bold lower and bold uppercase letters, respectively, e.g., $\mathbf{a}$ and $\mathbf{A}$. A mode-n vector of a tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ (with $\mathbb{K}$ meaning $\mathbb{R}$ or $\mathbb{C}$) is defined by fixing every index except the nth, e.g., $\mathbf{a}_{i_1 \ldots i_{n-1} : i_{n+1} \ldots i_N}$, and is a natural extension of a row or a column of a matrix. The mode-n unfolding of $\mathcal{A}$ is the matrix $\mathbf{A}_{(n)}$ with the mode-n vectors as its columns (following the ordering convention in [125]). An Mth-order slice of $\mathcal{A}$ is obtained by fixing all but M indices. The vectorization of $\mathcal{A}$, denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \ldots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1 + \sum_{k=1}^{N} (i_k - 1) J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$ (with $\prod_{m}^{k-1} (\cdot) = 1$ if $m > k-1$). The $\operatorname{unvec}(\cdot)$ operation is defined as the inverse of $\operatorname{vec}(\cdot)$.

The nth element in a sequence is indicated by a superscript between parentheses, e.g., $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$. The complex conjugate, transpose, conjugated transpose, inverse, and pseudoinverse are denoted as $\bar{\cdot}$, $\cdot^{\mathrm{T}}$, $\cdot^{\mathrm{H}}$, $\cdot^{-1}$ and $\cdot^{\dagger}$, respectively. A vector of length K with all entries equal to one is denoted as $\mathbf{1}_K$. The identity matrix of size $K \times K$ is denoted as $\mathbf{I}_K$. The binomial coefficient is denoted by $C_n^k = \frac{n!}{(n-k)!k!}$. $\mathbf{A} = \operatorname{diag}(\mathbf{a})$ is a diagonal matrix with the elements of $\mathbf{a}$ on the main diagonal.

The outer and Kronecker product are denoted by $\otimes$ and $\otimes$, respectively, and are related through a vectorization: $\operatorname{vec}(\mathbf{a} \otimes \mathbf{b}) = \mathbf{b} \otimes \mathbf{a}$. The mode-$n$ product of a tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $\mathbf{B} \in \mathbb{K}^{J_n \times I_n}$, denoted by $\mathcal{A} \cdot_n \mathbf{B} \in \mathbb{K}^{I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots I_N}$, is defined element-wise as $(\mathcal{A} \cdot_n \mathbf{B})_{i_1 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} a_{i_1 i_2 \ldots i_N} b_{j_n i_n}$. Hence, each mode-$n$ vector of the tensor $\mathcal{A}$ is multiplied with the matrix $\mathbf{B}$, i.e., $(\mathcal{A} \cdot_n \mathbf{B})_{(n)} = \mathbf{B}\mathbf{A}_{(n)}$. The inner product of two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by $\langle \mathcal{A}, \mathcal{B} \rangle$ and defined as $\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1}^{I_1} \sum_{i_2}^{I_2} \cdots \sum_{i_N}^{I_N} a_{i_1 i_2 \ldots i_N} \bar{b}_{i_1 i_2 \ldots i_N}$. The Khatri–Rao and Hadamard product are denoted by $\odot$ and $*$, respectively.

An Nth-order tensor has rank one if it can be written as the outer product of N nonzero vectors. The rank of a tensor is defined as the minimal number of rank-1 terms that generate the tensor as their sum. The mode-n rank of a tensor is defined as the rank of the mode-n unfolding. The multilinear rank of an Nth-order tensor is equal to the tuple of mode-n ranks.

3.1.2 Multilinear algebraic prerequisites

The CPD is a powerful model for various applications within signal processing, biomedical sciences, computer vision, data mining and machine learning [41], [125], [168].

Definition 20. A *polyadic decomposition* (PD) writes an Nth-order tensor

$\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ as a sum of R rank-1 terms:

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(N)} \overset{\text{def}}{=} \left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)} \right]\!\right],$$

in which the columns of the factor matrices $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$ are equal to the factor vectors $\mathbf{u}_r^{(n)}$ for $1 \leq r \leq R$. The PD is called *canonical* (CPD) if R is equal to the rank of $\mathcal{T}$, i.e., R is minimal.

The decomposition is *essentially unique* if it is unique up to trivial permutation of the rank-1 terms and scaling and counterscaling of the factors in the same rank-1 term. In the matrix case ($N = 2$) the CPD is not unique without additional assumptions for $R > 1$. Uniqueness is typically expected under rather mild conditions when $N > 2$; see, e.g., [70], [71], [73], [75], [128] and references therein.

The MLSVD of a higher-order tensor is a multilinear generalization of the SVD of a matrix [41], [54], [168].

Definition 21. A *multilinear singular value decomposition* (MLSVD) writes a tensor $\mathcal{T} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ as the product

$$\mathcal{T} = \mathcal{S} \cdot_1 \mathbf{U}^{(1)} \cdot_2 \mathbf{U}^{(2)} \cdots \cdot_N \mathbf{U}^{(N)} \overset{\text{def}}{=} \left[\!\left[\mathcal{S}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)} \right]\!\right]. \tag{3.1}$$

The factor matrices $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times I_n}$, for $1 \leq n \leq N$, are unitary matrices and the core tensor $\mathcal{S} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ is ordered and all-orthogonal [54].

The (truncated) MLSVD is a powerful tool in various applications such as compression, dimensionality reduction, and face recognition [59], [125], [204]. The decomposition is related to the LMLRA and the Tucker decomposition (TD); see [54], [214] and references therein. The mode-n unfolding of (3.1) is given by:

$$\mathbf{T}_{(n)} = \mathbf{U}^{(n)} \mathbf{S}_{(n)} \left(\mathbf{U}^{(N)} \otimes \cdots \otimes \mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n-1)} \otimes \cdots \otimes \mathbf{U}^{(1)} \right)^{\mathrm{T}}.$$

3.2 Linear systems with a CPD constrained solution

First, we define linear systems with a CPD constrained solution in subsection 3.2.1. Next, we discuss structure of the coefficient matrix and generic uniqueness in subsections 3.2.2 and 3.2.3, respectively.

3.2.1 Definition

In this chapter, linear systems of equations of which the solution can be represented by a tensor decomposition are considered. We limit ourselves to linear systems with a CPD structured solution, abbreviated as LS-CPD, but one can also use other decompositions such as the MLSVD, TT or HT [54], [96], [152]. Concretely, consider a linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ with coefficient matrix $\mathbf{A} \in \mathbb{K}^{M \times K}$, solution vector $\mathbf{x} \in \mathbb{K}^{K}$, and right-hand side $\mathbf{b} \in \mathbb{K}^{M}$. As such, we define LS-CPD as

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{with} \quad \mathbf{x} = \operatorname{vec}\left(\left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}\right]\!\right]\right) \tag{3.2}$$

with $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$, for $1 \leq n \leq N$, and $K = \prod_{n=1}^{N} I_n$. Equation (3.2) can be interpreted as a decomposition of a tensor $\mathcal{X} = \operatorname{unvec}(\mathbf{x})$ that is only *implicitly* known via the solution $\mathbf{x}$ of a linear system. Rather than K variables, the CPD structure allows the vector $\mathbf{x}$ of length K to be represented by only $\mathcal{O}(RI')$ variables with $I' = \sum_{n=1}^{N} I_n$, or, when accommodating for scaling indeterminacies, $R(I' - N + 1)$ free variables. For example, consider the following second-order rank-1 structure $[x; y; z] \otimes [u; v; w]$ which is equivalent with $[1; y/x; z/x] \otimes [ux; vx; wx]$, reducing the number of variables by one. For higher-order structures, this extends to a reduction by $N - 1$, i.e., from $\mathcal{O}\left(I^N\right)$ to $\mathcal{O}\left(NI\right)$ entries. This compact representation allows one to solve the structured linear system in (3.2) in the underdetermined case ($M < K$), enabling a compressed-sensing-style approach [34], [78].

We show that LS-CPDs are multilinear systems of equations. Let $\mathcal{A}$ be a tensor of order $N+1$ with dimensions $M \times I_1 \times I_2 \times \cdots \times I_N$ such that its mode-1 unfolding $\mathbf{A}_{(1)}$ equals the coefficient matrix $\mathbf{A}$, i.e., we have $\mathbf{A}_{(1)} = \mathbf{A}$. We can then rewrite (3.2) as a set of inner products:

$$\left\langle \mathcal{A}_m, \left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}\right]\!\right] \right\rangle = b_m, \text{ for } 1 \leq m \leq M, \tag{3.3}$$

in which $\mathcal{A}_m = \mathcal{A}(m, :, :, \ldots, :)$ is the Nth-order "horizontal slice" of $\mathcal{A}$. If $N = R = 1$, we obtain a linear system of equations and (3.3) reduces to:

$$\langle \mathbf{a}_m^{\mathrm{T}}, \mathbf{x} \rangle = b_m, \quad \text{for} \quad 1 \leq m \leq M,$$

with $\mathbf{a}_m^{\mathrm{T}}$ the mth row of $\mathbf{A}$. Clearly, (3.3) is a set of M multilinear equations. For example, consider the following simple LS-CPD with $N = 2$ and $R = 1$:

$$\mathbf{A}\operatorname{vec}(\mathbf{u} \otimes \mathbf{v}) = \mathbf{b}, \quad \text{or, equivalently,} \quad \mathbf{A}(\mathbf{v} \otimes \mathbf{u}) = \mathbf{b} \tag{3.4}$$

with $\mathbf{A} = \mathbf{A}_{(1)} \in \mathbb{K}^{M \times IJ}$, $\mathbf{u} \in \mathbb{K}^{I}$, and $\mathbf{v} \in \mathbb{K}^{J}$. Equation (3.4) is clearly a compact form of the following set of multilinear equations (with values

$I = J = 2$ and $M = 4$):

$$\begin{cases} a_{111}v_1u_1 + a_{121}v_1u_2 + a_{112}v_2u_1 + a_{122}v_2u_2 = b_1, \\ a_{211}v_1u_1 + a_{221}v_1u_2 + a_{212}v_2u_1 + a_{222}v_2u_2 = b_2, \\ a_{311}v_1u_1 + a_{321}v_1u_2 + a_{312}v_2u_1 + a_{322}v_2u_2 = b_3, \\ a_{411}v_1u_1 + a_{421}v_1u_2 + a_{412}v_2u_1 + a_{422}v_2u_2 = b_4. \end{cases}$$

or, equivalently, we have $\mathcal{A} \cdot_1 \mathbf{u}^{\mathrm{T}} \cdot_2 \mathbf{v}^{\mathrm{T}} = \mathbf{b}$.

3.2.2 LS-CPD as CPD by exploiting structure of A

For particular types of structure on the coefficient matrix $\mathbf{A}$ in (3.2), the LS-CPD problem can be reformulated as a (constrained) tensor decomposition. Two examples are investigated here. First, if the coefficient matrix in (3.2) is a diagonal matrix $\mathbf{D} = \mathrm{diag}(\mathbf{d})$, the LS-CPD model reduces to a *weighted* CPD of a tensor $\mathcal{B} = \mathrm{unvec}(\mathbf{b})$ [154], [189], i.e., we have:

$$\mathcal{B} = \mathcal{D} * \left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}\right]\!\right]$$

with $\mathcal{D}$ a tensor defined such that $\mathcal{D} = \mathrm{unvec}(\mathbf{d})$. This model can also be used to handle missing entries by setting the corresponding weights to zero [3], [214]. It is clear that an LS-CPD reduces to a CPD if $\mathbf{D}$ is the identity matrix.

Next, we consider a coefficient matrix $\mathbf{A} \in \mathbb{K}^{M \times K}$ that has a Kronecker product structure: $\mathbf{A} = \mathbf{A}^{(N)} \otimes \mathbf{A}^{(N-1)} \otimes \cdots \otimes \mathbf{A}^{(1)}$ with $\mathbf{A}^{(n)} \in \mathbb{K}^{J_n \times I_n}$ such that $M = \prod_{n=1}^{N} J_n$ and $K = \prod_{n=1}^{N} I_n$. Note that

$$\mathrm{vec}\left(\left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}\right]\!\right]\right) = \left(\mathbf{U}^{(N)} \odot \mathbf{U}^{(N-1)} \odot \cdots \odot \mathbf{U}^{(1)}\right) \mathbf{1}_R.$$

One can then show that (3.2) can be written as [136]:

$$\begin{aligned} \left(\mathbf{A}^{(N)} \otimes \mathbf{A}^{(N-1)} \otimes \cdots \otimes \mathbf{A}^{(1)}\right) \left(\mathbf{U}^{(N)} \odot \mathbf{U}^{(N-1)} \odot \cdots \odot \mathbf{U}^{(1)}\right) \mathbf{1}_R &= \mathbf{b}, \\ \left(\mathbf{A}^{(N)}\mathbf{U}^{(N)} \odot \mathbf{A}^{(N-1)}\mathbf{U}^{(N-1)} \odot \cdots \odot \mathbf{A}^{(1)}\mathbf{U}^{(1)}\right) \mathbf{1}_R &= \mathbf{b}, \\ \mathrm{vec}\left(\left[\!\left[\mathbf{A}^{(1)}\mathbf{U}^{(1)}, \mathbf{A}^{(2)}\mathbf{U}^{(2)}, \ldots, \mathbf{A}^{(N)}\mathbf{U}^{(N)}\right]\!\right]\right) &= \mathbf{b}, \end{aligned}$$

which is equivalent to:

$$\left[\!\left[\mathbf{A}^{(1)}\mathbf{U}^{(1)}, \mathbf{A}^{(2)}\mathbf{U}^{(2)}, \ldots, \mathbf{A}^{(N)}\mathbf{U}^{(N)}\right]\!\right] = \mathcal{B}. \tag{3.5}$$

Expression (3.5) is a CPD with linear constraints on the factor matrices and is also known as the CANDELINC model [35], [125]; note that compatibility of the dimensions of $\mathbf{U}^{(n)}$ and $\mathbf{A}^{(n)}$ is essential to reformulate the LS-CPD as (3.5). Expression (3.5) can be computed using projection or by using a specific algorithm if the tensor $\mathcal{B}$ has missing entries [212].

3.2.3 Generic uniqueness

We show that generic uniqueness is possible when the number of equations is larger than the number of free variables plus one. More specifically, we present a bound on M guaranteeing uniqueness of $\mathbf{x}$ in (3.2) for a generic $M \times K$ coefficient matrix $\mathbf{A}$. Generic uniqueness means that we have uniqueness with probability one when the entries of $\mathbf{A}$ are drawn from absolutely continuous probability density functions. We refer the reader to [70], [71], [73], [75], [128] and references therein regarding (generic) uniqueness conditions for the factor matrices in the CPD of $\mathcal{X}$. Our main result states that in order to have a generically unique solution, we need at least as many equations as free variables (i.e., after removing scaling indeterminacies) plus one.

Lemma 1. Let $\mathbf{A}$ be a generic $M \times K$ matrix with $K = I_1 \cdots I_N$. Define $\mathbf{b} = \mathbf{A}\mathrm{vec}(\mathcal{X}_0)$ with $\mathcal{X}_0$ a $I_1 \times \cdots \times I_N$ tensor with rank less than or equal to R. In that case, the solution vector $\mathbf{x}$ in (3.2) is unique if $M \geq (I_1 + \cdots + I_N - N + 1)R + 1$.

Proof. Consider an irreducible algebraic variety $V \in \mathbb{K}^K$ of dimension d_V. It is known that a generic plane of dimension less than or equal to $K - d_V - 1$ does not intersect with V [176, Theorem A.8.1, p. 326]. It is clear that a generic plane of dimension $K - M$ can be interpreted as the null space of a generic $M \times K$ matrix. Hence, if $\mathbf{A}$ is a generic $M \times K$ matrix, $\mathbf{v}_0 \in V$ and $\mathbf{b} := \mathbf{A}\mathbf{v}_0$, then the problem

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad \text{with} \quad \mathbf{x} \in V \tag{3.6}$$

has a unique solution whenever $K - M \leq K - d_V - 1$ or $M \geq d_V + 1$. We interpret (3.2) as (3.6) in which V is the Zariskii closure of the set of $I_1 \times \cdots \times I_N$ tensors whose rank does not exceed R. Since a generic tensor in V can be parameterized with at most $(I_1 + \cdots + I_N - N + 1)R$ parameters, it follows that $d_V \geq (I_1 + \cdots + I_N - N + 1)R$. Hence, a solution vector $\mathbf{x}$ in (3.2) is unique if $M \geq d_V + 1 \geq (I_1 + \cdots + I_N - N + 1)R + 1$. □

3.3 Algorithms

First, we derive an algebraic method to solve an LS-CPD with $R = 1$ in subsection 3.3.1. Next, we develop an optimization-based algorithm for general LS-CPDs in subsection 3.3.2.

3.3.1 Algebraic computation

We present an algebraic method to solve (3.2). The derivation is closely related to [47]. Importantly, all steps can be performed by means of conventional linear algebra. The overall algebraic procedure is summarized in Algorithm 3.1. This method finds the exact solution in the case of exact problems, but can also be used to obtain a good initialization for optimization-based methods in the case of perturbed problems.

It is well-known that a tensor $\mathcal{X}$ of order N has rank one if and only if all its matrix unfoldings have rank one, i.e., we have:

$$\operatorname{rank}\left(\mathbf{X}_{(n)}\right) = R = 1, \quad \text{for } 1 \leq n \leq N. \tag{3.7}$$

In this particular case a solution $\mathbf{x}$ to (3.2) is also a solution of

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{with} \quad \mathbf{x} = \operatorname{vec}(\mathcal{X}), \text{ where } \mathcal{X} \text{ satisfies (3.7)} \tag{3.8}$$

and a solution to (3.8) is also a solution to (3.2). The case $R > 1$ relates to linear systems with a MLSVD constrained solution; see [22]. We can compute a solution of (3.2) *algebraically* in two steps as follows. First, we use (3.8) to recover $\mathcal{X}$. Next, we compute the (exact) rank-1 CPD of $\mathcal{X}$.

Trivial case

Assume that the solution of the unstructured linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is unique, i.e., the null space of the extended matrix $\begin{bmatrix}\mathbf{A} & \mathbf{b}\end{bmatrix}$ is one-dimensional. In that case, we can compute the solution to (3.8) by ignoring the multilinear structure (3.7), i.e., we solve for $\mathbf{x}$ and subsequently compute a CPD of $\mathcal{X} = \operatorname{unvec}(\mathbf{x})$. Clearly, the tensor $\mathcal{X}$ is unique if $\mathbf{b} \neq \mathbf{0}$ or is unique up to a scaling factor if $\mathbf{b} = \mathbf{0}$. This approach is the naive method that we have mentioned in section 3.1.

Reduction of the general case to the trivial case

We explain how to find a solution of (3.8) when the dimension of the null space of $\begin{bmatrix}\mathbf{A} & \mathbf{b}\end{bmatrix}$ is larger than one, e.g., when $\mathbf{A}$ is a fat matrix or rank-deficient. We limit ourselves to the case where $\mathbf{b} \neq \mathbf{0}$, which implies that the dimension of the null space of $\mathbf{A}$ is at least one. It can be shown that the case where $\mathbf{b} = \mathbf{0}$ follows in a similar way.

Let $\mathbf{f}^{(0)}$ be a particular solution of $\mathbf{A}\mathbf{x} = \mathbf{b}$ and let the vectors $\mathbf{f}^{(l)}$, for $1 \leq l \leq L$, form a basis of the L-dimensional null space of $\mathbf{A}$. Consider the tensorized versions of $\mathbf{f}^{(l)}$ denoted by $\mathcal{F}^{(l)} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$, for $0 \leq l \leq L$. In order to solve (3.8), we have to find values c_l, for $1 \leq l \leq L$, such that

$\mathcal{X} = \mathcal{F}^{(0)} + c_1\mathcal{F}^{(1)} + \cdots + c_L\mathcal{F}^{(L)}$ (with $c_0 = 1$) satisfies (3.7), i.e., we have:

$$\operatorname{rank}\left(\mathbf{X}_{(n)}\right) = \operatorname{rank}\left(\mathbf{F}^{(0)}_{(n)} + c_1\mathbf{F}^{(1)}_{(n)} + \cdots + c_L\mathbf{F}^{(L)}_{(n)}\right) = 1, \quad 1 \leq n \leq N. \tag{3.9}$$

We can reformulate (3.9) as the following LS-CPD problem:

$$\tilde{\mathbf{A}}(\mathbf{c} \otimes \mathbf{c}) = \mathbf{0} \quad \text{with} \quad \mathbf{c} = [1\ c_1\ \cdots\ c_L]^{\mathrm{T}} \tag{3.10}$$

with, as explained below, $\tilde{\mathbf{A}}$ constructed from the tensors $\mathcal{F}^{(l)}$ such that each row of $\tilde{\mathbf{A}}$ is a vectorized $(L+1) \times (L+1)$ symmetric matrix. We make the assumption that the intersection of the null space of $\tilde{\mathbf{A}}$ with the subspace of vectorized symmetric matrices is one-dimensional. In practice this assumption is satisfied when the difference between the number of rows and columns of $\tilde{\mathbf{A}}$ is sufficiently large. In that case, the solution $\mathbf{c} \otimes \mathbf{c}$ is unique and can be computed as explained for the trivial case, from which $\mathbf{c}$ can be easily recovered.

We explain the construction of $\tilde{\mathbf{A}}$ in more detail. First, partition $\tilde{\mathbf{A}}$ as follows:

$$\tilde{\mathbf{A}} = \left[\tilde{\mathbf{A}}^{(1)^{\mathrm{T}}} \quad \tilde{\mathbf{A}}^{(2)^{\mathrm{T}}} \quad \cdots \quad \tilde{\mathbf{A}}^{(N)^{\mathrm{T}}}\right]^{\mathrm{T}}, \tag{3.11}$$

where the matrices $\tilde{\mathbf{A}}^{(n)}$ correspond to the constraints in (3.9). Consider the following definition.

Definition 22. The second compound matrix $\mathcal{C}_2(\mathbf{F})$ of an $I \times J$ matrix $\mathbf{F}$, with $2 \leq \min(I, J)$, is a $C_I^2 \times C_J^2$ matrix containing all 2×2 minors of $\mathbf{F}$ ordered lexicographically [108].

It is well-known that the following algebraic identity holds for any 2×2 matrices $\mathbf{F}^{(0)}, \ldots, \mathbf{F}^{(L)}$ and values $c_0, \ldots, c_L$:

$$\begin{aligned}&\det(c_0\mathbf{F}^{(0)} + c_1\mathbf{F}^{(1)} + \cdots + c_L\mathbf{F}^{(L)}) = \\ &\frac{1}{2}\sum_{j_1,j_2=1}^{L+1} c_{j_1}c_{j_2}\left[\det(\mathbf{F}^{(j_1)} + \mathbf{F}^{(j_2)}) - \det(\mathbf{F}^{(j_1)}) - \det(\mathbf{F}^{(j_2)})\right].\end{aligned} \tag{3.12}$$

By applying (3.12) to each 2×2 submatrix of $c_0\mathbf{F}^{(0)}_{(n)} + c_1\mathbf{F}^{(1)}_{(n)} + \cdots + c_L\mathbf{F}^{(L)}_{(n)}$, we obtain:

$$\begin{aligned}&\mathcal{C}_2\left(c_0\mathbf{F}^{(0)}_{(n)} + c_1\mathbf{F}^{(1)}_{(n)} + \cdots + c_L\mathbf{F}^{(L)}_{(n)}\right) = \\ &\frac{1}{2}\sum_{j_1,j_2=1}^{L+1} c_{j_1}c_{j_2}\left[\mathcal{C}_2\left(\mathbf{F}^{(j_1)}_{(n)} + \mathbf{F}^{(j_2)}_{(n)}\right) - \mathcal{C}_2\left(\mathbf{F}^{(j_1)}_{(n)}\right) - \mathcal{C}_2\left(\mathbf{F}^{(j_2)}_{(n)}\right)\right].\end{aligned} \tag{3.13}$$

Condition (3.9) states that all 2×2 minors of the matrix $\mathbf{X}_{(n)} = \mathbf{F}^{(0)}_{(n)} + c_1\mathbf{F}^{(1)}_{(n)} + \cdots + c_L\mathbf{F}^{(L)}_{(n)}$ are zero, or, in other words, we have that $\mathcal{C}_2\left(\mathbf{X}_{(n)}\right) = \mathbf{0}$.

Hence, according to (3.13), we have:

$$\sum_{j_1,j_2=1}^{L+1} c_{j_1}c_{j_2}\left[\mathcal{C}_2\left(\mathbf{F}_{(n)}^{(j_1)}+\mathbf{F}_{(n)}^{(j_2)}\right)-\mathcal{C}_2\left(\mathbf{F}_{(n)}^{(j_1)}\right)-\mathcal{C}_2\left(\mathbf{F}_{(n)}^{(j_2)}\right)\right]=0,$$

with $c_0 = 1$, which is equivalent to

$$\tilde{\mathbf{A}}^{(n)}(\mathbf{c}\otimes\mathbf{c})=\mathbf{0},\quad \text{with}\quad \mathbf{c}=[1\ c_1\ \dots\ c_L]^{\mathrm{T}},\ 1\leq n\leq N,$$

in which $\tilde{\mathbf{A}}^{(n)}$ has size $C_{I_n}^2 C_{KI_n^{-1}}^2 \times (L+1)^2$ and is defined column-wise as follows:

$$\tilde{\mathbf{a}}_{j_2+(L+1)(j_1-1)}^{(n)}=\operatorname{vec}\left(\mathcal{C}_2\left(\mathbf{F}_{(n)}^{(j_1)}+\mathbf{F}_{(n)}^{(j_2)}\right)-\mathcal{C}_2\left(\mathbf{F}_{(n)}^{(j_1)}\right)-\mathcal{C}_2\left(\mathbf{F}_{(n)}^{(j_2)}\right)\right). \tag{3.14}$$

In Algorithm 3.1, the number of rows of $\tilde{\mathbf{A}}$ should be at least the dimension of the subspace of the symmetric $L+1\times L+1$ matrices minus one. Hence, a *necessary* condition for the algebraic computation is that $\sum_{n=1}^{N} C_{I_n}^2 C_{KI_n^{-1}}^2 \geq (L+1)(L+2)/2-1 \geq (K-M+1)(K-M+2)-1$. Note that L satisfies $L = K - \dim(\operatorname{range}(\mathbf{A})) \geq K-M$ by the rank nullity theorem. The computational complexity of Algorithm 3.1 is dominated by the construction of $\tilde{\mathbf{A}}$.

Algorithm 3.1: Algebraic algorithm to solve a linear system of equations $\mathbf{Ax}=\mathbf{b}$ in which the solution $\mathbf{x}$ has a rank-1 CPD structure.

```
1: Input: $\mathbf{A}$ and nonzero $\mathbf{b}$
2: Output: $\{\mathbf{u}^{(n)}\}_{n=1}^{N}$
3: Find $\mathbf{f}^{(0)} \in \mathbb{K}^K$ such that $\mathbf{Af}^{(0)} = \mathbf{b}$
4: Find $\mathbf{f}^{(l)} \in \mathbb{K}^K$, for $1 \leq l \leq L$, that form a basis for null($\mathbf{A}$) $\in \mathbb{K}^{K\times L}$
5: Reshape $\mathbf{f}^{(l)}$ into $I_1 \times I_2 \times \cdots \times I_N$ tensors $\mathcal{F}^{(l)}$, for $0 \leq l \leq L$
6: Construct $\tilde{\mathbf{A}}^{(1)},\dots,\tilde{\mathbf{A}}^{(N)}$ as in (3.14) and construct $\tilde{\mathbf{A}}$ as in (3.11)
7: Find a nonzero solution of $\tilde{\mathbf{A}}\tilde{\mathbf{c}} = 0$ (if null($\tilde{\mathbf{A}}$) is one-dimensional)
8: Find the vector $\mathbf{c} = [1\ c_1\ \dots\ c_L]^T$ such that $\mathbf{c}\otimes\mathbf{c}$ is proportional to $\tilde{\mathbf{c}}$
9: Construct $\mathcal{X} = \mathcal{F}^{(0)} + c_1\mathcal{F}^{(1)} + \cdots + c_L\mathcal{F}^{(L)}$
10: Compute the rank-1 CPD of $\mathcal{X} = [\![\mathbf{u}^{(1)},\mathbf{u}^{(2)},\dots,\mathbf{u}^{(N)}]\!]$
```

3.3.2 Optimization-based methods

In this subsection, we solve the LS-CPD problem in (3.2) via a least-squares approach, leading to the following optimization problem:

$$\min_{\mathbf{z}} f = \frac{1}{2}\left|\left|\mathbf{r}(\mathbf{z})\right|\right|_{\mathrm{F}}^2 \tag{3.15}$$

in which the residual $\mathbf{r}(\mathbf{z}) \in \mathbb{K}^M$ is defined as

$$\mathbf{r}(\mathbf{z}) = \mathbf{A}\mathrm{vec}\left(\left[\!\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}\right]\!\!\right]\right) - \mathbf{b},$$

where we have concatenated the optimization variables in a vector $\mathbf{z} \in \mathbb{K}^{RI'}$ with $I' = \sum_{n=1}^{N} I_n$ as $\mathbf{z} = \left[\mathrm{vec}\left(\mathbf{U}^{(1)}\right); \ \mathrm{vec}\left(\mathbf{U}^{(2)}\right); \ \cdots; \ \mathrm{vec}\left(\mathbf{U}^{(N)}\right)\right]$. To solve the NLS problem (3.15), we use the Gauss–Newton (GN) method which is a particular NLS algorithm [150]. In order to attain global convergence, we employ a trust region approach, allowing the algorithm to converge to a (local) minimum for any starting point under mild conditions. Notably, if the algorithm converges, GN often converges quadratically, hence, fewer iterations are needed. The GN algorithm requires expressions for the objective function, gradient, Gramian, and Gramian-vector product. Although we focus on the GN method, the expressions can be used to implement other NLS algorithms as well as qN algorithms. In order to implement the methods, we use the complex optimization framework from [177], [179] which provides implementations for qN and NLS algorithms as well as line search, plane search, and trust region methods.

The GN method solves (3.15) by linearizing the residual vector $\mathbf{r}(\mathbf{z})$ and solving a least-squares problem in each iteration k:

$$\min_{\mathbf{p}_k} \frac{1}{2} ||\mathbf{r}(\mathbf{z}_k) + \mathbf{J}_k\mathbf{p}_k||_{\mathrm{F}}^2 \quad \text{s.t.} \quad ||\mathbf{p}_k|| \leq \Delta_k$$

with step $\mathbf{p}_k = \mathbf{z}_{k+1} - \mathbf{z}_k$ and trust-region radius Δ_k [150]. The Jacobian $\mathbf{J} = \mathrm{d}\mathbf{r}(\mathbf{z})/\mathrm{d}\mathbf{z} \in \mathbb{K}^{M \times RI'}$ is evaluated at $\mathbf{z}_k$. The exact solution to the linearized problem is given by the normal equations:

$$\mathbf{J}_k^{\mathrm{H}}\mathbf{J}_k\mathbf{p}_k = -\mathbf{J}_k^{\mathrm{H}}\mathbf{r}(\mathbf{z}_k), \quad \text{or, equivalently,} \quad \mathbf{H}_k\mathbf{p}_k = -\overline{\mathbf{g}}_k. \tag{3.16}$$

In the NLS formulation, $\mathbf{H} \in \mathbb{K}^{RI' \times RI'}$ is the Gramian of the Jacobian which is an approximation to the Hessian of f [150]. The conjugated gradient $\overline{\mathbf{g}} \in \mathbb{K}^{RI'}$ is defined as $\overline{\mathbf{g}} = (\partial f / \partial \mathbf{z})^{\mathrm{H}}$. The normal equations are solved inexactly using several preconditioned conjugate gradient (CG) iterations to reduce the computational complexity. After solving (3.16), the variables can be updated as $\mathbf{z}_{k+1} = \mathbf{z}_k + \mathbf{p}_k$. While a dogleg trust-region method is used here, other updating methods such as line and plane search can be used as well, see [150] for details. In the remainder of this subsection we derive the required expressions for the GN method summarized in Algorithm 3.2.

Objective function

We evaluate the objective function f by taking the sum of squared entries of the residual $\mathbf{r}(\mathbf{z})$. The latter can be computed by using contractions as

Algorithm 3.2: LS-CPD using Gauss–Newton with dogleg trust region.

```
Input:  $\mathbf{A}$, $\mathbf{b}$, and initial estimate for $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$
Output:  $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$
while not converged do
    Compute gradient $\mathbf{g}$ using (3.17).
    Use PCG to solve $\mathbf{H}\mathbf{p} = -\overline{\mathbf{g}}$ for $\mathbf{p}$ using Gramian-vector products as in (3.20)
    using a (block)-Jacobi preconditioner, see subsection 3.3.2.
    Update $\mathbf{U}^{(n)}$, for $1 \leq n \leq N$, using dogleg trust region from $\mathbf{p}$, $\mathbf{g}$, and function
    evaluation (3.15).
end while
```

follows:

$$\mathbf{r}(\mathbf{z}) = \sum_{r=1}^{R} \mathcal{A} \cdot_2 \mathbf{u}_r^{(1)^{\mathrm{T}}} \cdot_3 \mathbf{u}_r^{(2)^{\mathrm{T}}} \cdots \cdot_{N+1} \mathbf{u}_r^{(N)^{\mathrm{T}}} - \mathbf{b}.$$

Gradient

We partition the gradient as

$$\mathbf{g} = \left[\mathbf{g}^{(1,1)}; \mathbf{g}^{(1,2)}; \ldots; \mathbf{g}^{(1,R)}; \mathbf{g}^{(2,1)}; \ldots; \mathbf{g}^{(N,R)}\right]$$

in which the subgradients $\mathbf{g}^{(n,r)} \in \mathbb{K}^{I_n}$ are defined as

$$\mathbf{g}^{(n,r)} = \left(\mathbf{J}^{(n,r)}\right)^{\mathrm{T}} \overline{\mathbf{r}(\mathbf{z})} \tag{3.17}$$

in which $\mathbf{J}^{(n,r)}$ is defined as

$$\mathbf{J}^{(n,r)} = \frac{\partial \mathbf{r}(\mathbf{z})}{\partial \mathbf{u}_r^{(n)}} = \left(\mathcal{A} \cdot_2 \mathbf{u}_r^{(1)^{\mathrm{T}}} \cdots \cdot_n \mathbf{u}_r^{(n-1)^{\mathrm{T}}} \cdot_{n+2} \mathbf{u}_r^{(n+1)^{\mathrm{T}}} \cdots \cdot_{N+1} \mathbf{u}_r^{(N)^{\mathrm{T}}}\right)_{(1)}. \tag{3.18}$$

Equation (3.18) equals the (n, r)th sub-Jacobian, using a similar partitioning for $\mathbf{J}$. The sub-Jacobians require a contraction in all but the first and nth mode and are precomputed.

Gramian of the Jacobian

We partition the Gramian $\mathbf{H}$ into a grid of $NR \times NR$ blocks $\mathbf{H}^{(n,r,m,l)}$ with $1 \leq n, m \leq N$ and $1 \leq r, l \leq R$. Each block $\mathbf{H}^{(n,r,m,l)}$ is defined by:

$$\mathbf{H}^{(n,r,m,l)} = \left(\mathbf{J}^{(n,r)}\right)^{\mathrm{H}} \mathbf{J}^{(m,l)}, \tag{3.19}$$

using the sub-Jacobians in (3.18). Equation (3.19) approximates the second-order derivative of f with respect to the variables $\mathbf{u}_r^{(n)}$ and $\mathbf{u}_l^{(m)}$.

Table 3.1: The per-iteration computational complexity of the NLS algorithm for LS-CPD is dominated by the computation of the Jacobian. *The algorithm uses a trust region approach to determine the update, which requires* it_{TR} *additional evaluations of the objective function.*

	Calls per iteration	Complexity
Objective function	$1 + \text{it}_{\text{TR}}$	$\mathcal{O}(MRI^N)$
Jacobian	1	$\mathcal{O}(MRNI^N)$
Gradient	1	$\mathcal{O}(MRNI)$
Gramian	1	$\mathcal{O}(MR^2N^2I^2)$
Gramian-vector	it_{CG}	$\mathcal{O}(MRNI)$

As preconditioned CG (PCG) is used, only matrix vector-products are needed. The full Gramian is never constructed because one can exploit the block structure to compute fast matrix-vector products. Hence, in each iteration we compute Gramian-vector products of the form $\mathbf{z} = \mathbf{J}^{\mathrm{H}}\mathbf{J}\mathbf{y}$ as follows:

$$\mathbf{z}^{(n,r)} = \left(\mathbf{J}^{(n,r)}\right)^{\mathrm{H}} \left(\sum_{n=1}^{N}\sum_{r=1}^{R}\mathbf{J}^{(n,r)}\mathbf{y}^{(n,r)}\right), \text{ for } 1 \leq n \leq N, \text{ and } 1 \leq r \leq R, \tag{3.20}$$

in which we partitioned $\mathbf{z}$ and $\mathbf{y}$ in a similar way as before.

In this chapter, we use either a block-Jacobi or Jacobi preconditioner to improve convergence or reduce the number of CG iterations. In the former case, we compute the $(I_n \times I_n)$ Gramians $\mathbf{H}^{(n,n,r,r)}$, for $1 \leq n \leq N$ and $1 \leq r \leq R$, *and* their inverses in each iteration. Combining both operations leads to a per-iteration computational complexity of $\mathcal{O}(MI_n^2 + I_n^3)$ which is relatively expensive, especially for large problems. One can instead use a Jacobi preconditioner which uses a diagonal approximation of the Gramian and, consequently, an inexpensive computation of the inverse. The diagonal elements are computed as the inner product $\mathbf{J}_{i_n}^{(n,r)^{\mathrm{H}}}\mathbf{J}_{i_n}^{(n,r)}$, for $1 \leq i_n \leq I_n$, leading to an overall computational complexity of $\mathcal{O}(MI_n + I_n)$ which is relatively inexpensive. We compare the effectiveness of the Jacobi and block-Jacobi preconditioner in section 3.4.

Complexity

We report the per-iteration complexity of the NLS algorithm for LS-CPD in Table 3.1 by counting multiplications. For simplicity, we assume that $I_1 = I_2 = \cdots = I_N = I$ in (3.2). Clearly the computational complexity is dominated by the computation of the sub-Jacobians in (3.18). The computational complexity can be reduced by computing the contractions in (3.18) as efficiently as possible. Note that the evaluation of the objective function is a factor N less expensive.

Efficient contractions

The per-iteration complexity of the NLS algorithm is relatively high, however, only a few iterations are often necessary in order to obtain convergence. One can reduce the overall computation time of the algorithm by reducing the computational cost per iteration or the number of iterations. We have shown that the computation of the Jacobians is relatively expensive; see Table 3.1. Computing the sub-Jacobians requires the sequential computation of $N-1$ contractions of the form $\mathcal{A} \cdot_n \mathbf{x}^{(n)^{\mathrm{T}}} \in \mathbb{K}^{I_1 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_N}$ which are defined as

$$(\mathcal{A} \cdot_n \mathbf{x}^{(n)^{\mathrm{T}}})_{i_1 \ldots i_{n-1} i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} a_{i_1 \ldots i_N} x_{i_n}^{(n)} \tag{3.21}$$

with $\mathcal{A} \in \mathbb{K}^{I_1 \times \cdots \times I_N}$ and a vector $\mathbf{x}^{(n)} \in \mathbb{K}^{I_n}$. Clearly, it is important to perform the contractions as efficiently as possible to reduce the per-iteration complexity of the algorithm. Note that the computation of the contractions can be done in a memory-efficient way by computing the contractions sequentially via the matrix unfoldings and permuting the first mode of $\mathcal{A}$ to the middle. This approach guarantees that $\mathcal{A}$ is permuted in memory at most once.

One way to compute contractions efficiently is by exploiting all possible structure of the coefficient tensor $\mathcal{A}$ in (3.21). For example, if $\mathbf{A}$ is the identity matrix, the LS-CPD problem reduces to a CPD. In that case, the Gramians and their inverses can be computed efficiently by storing the Gramians of the factor matrices; see [180]. If $\mathbf{A}$ has a Kronecker product structure, the LS-CPD problem reduces to a CANDELINC model, as shown in subsection 3.2.2, which can be computed efficiently in both the dense and sparse case; see [125], [212]. For specific types of structure in $\mathbf{A}$, forward-adjoint oracles [67] can be generalized to the multilinear case.

Let us illustrate how we can compute efficient contractions in the case of a sparse $\mathcal{A}$. Assume we have a vector $\mathbf{a} \in \mathbb{K}^M$ containing the M nonzero values of $\mathcal{A}$ and corresponding index sets $\mathcal{I}_m = \{i_1^{(m)}, i_2^{(m)}, \ldots, i_N^{(m)}\}$, for $1 \leq m \leq M$. We can then compute (3.21) efficiently as $\mathbf{w} = \mathbf{a} * \mathbf{x}^{(n)}_{\mathcal{I}'_n}$ with $\mathcal{I}'_n = \{i_n^{(1)}, i_n^{(2)}, \ldots, i_n^{(M)}\}$, for $1 \leq n \leq N$. As such, we obtain a new index-value pair with $\mathbf{w} \in \mathbb{K}^M$ and $\mathcal{J}_m = \mathcal{I}_m \backslash i_n^{(m)}$ for $1 \leq m \leq M$.

3.4 Numerical experiments

First, two proof-of-concept experiments are conducted to illustrate the algebraic and optimization-based methods in subsection 3.4.1. Next, we compare accuracy and time complexity of the naive, algebraic, and NLS method in subsection 3.4.2. We also compare algebraic and random initialization methods for the NLS algorithm in subsection 3.4.3. In subsection 3.4.4, we compare

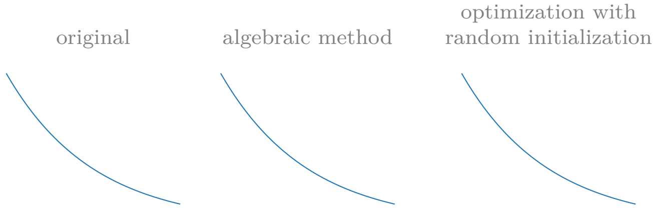


Figure 3.2: Our algebraic method and optimization-based method (with random initialization) can perfectly reconstruct an exponential solution vector in the noiseless case.

the Jacobi and block-Jacobi preconditioner for the NLS algorithm. All computations are done with Tensorlab [215]. We define the relative error $\epsilon_{\mathbf{x}}$ as the relative difference in Frobenius norm $\|\mathbf{x}-\hat{\mathbf{x}}\|_{\mathrm{F}}/\|\mathbf{x}\|_{\mathrm{F}}$ with $\hat{\mathbf{x}}$ being an estimate for $\mathbf{x}$. We use factor matrices in which the elements are drawn from the standard normal distribution, unless stated otherwise, to generate tensors. In that case the factor matrices are well-conditioned because the expected angle between the factor vectors is 90° for large matrices. The coefficient matrices $\mathbf{A}$ are constructed in a similar way, unless stated otherwise. We use i.i.d. Gaussian noise to perturb the entries of a tensor, unless stated otherwise. The noise is scaled to obtain a given signal-to-noise ratio (SNR) (with the signal equal to the noiseless tensor). If we consider a perturbed LS-CPD problem, we perturb the right-hand side in (3.2), unless stated otherwise.

3.4.1 Proof-of-concept

We give two simple proof-of-concept experiments, illustrating our algorithms for linear systems with a solution that can be represented or well approximated by a low-rank model. First, consider an LS-CPD with a solution $\mathbf{x} \in \mathbb{K}^K$ that is constrained to be an exponential, i.e., $x_k = e^{-2k}$ evaluated in K equidistant samples in $[0, 1]$. It is known that sums of exponentials can be exactly represented by low-rank tensors [20], [21], [51]. In this case, the corresponding tensor $\mathcal{X} = \operatorname{unvec}(\mathbf{x})$ has rank one ($R = 1$) [51]. We choose $N = 3$, $I_1 = I_2 = I_3 = I = 4$, $K = I^3 = 64$, and $M = 34$. We compute a solution using the algebraic method and the NLS algorithm with random initialization. Perfect reconstruction of the exponential is obtained with both methods as shown in Figure 3.2.

In the previous experiment the solution vector could be exactly represented by a low-rank tensor. Many signals, such as Gaussians, rational functions, and periodic signals, can also be well approximated by a low-rank model [20], [21]. In this experiment, we consider an LS-CPD of which the solution vector

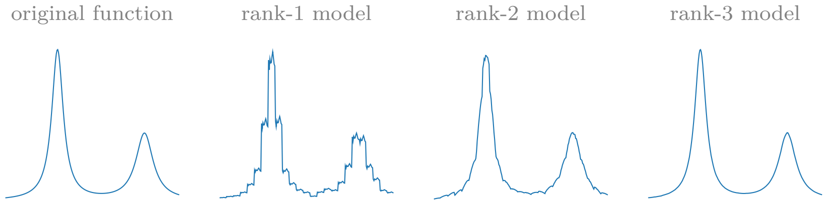


Figure 3.3: Our optimization-based method (with random initialization) can reconstruct the rational solution vector in the noiseless case. *Increasing the rank of the CPD model improves the accuracy of the solution. For example, the rank-3 model is almost indistinguishable from the original function.*

$\mathbf{x} = \operatorname{vec}(\mathbf{X})$ is a rational function:

$$x_k = \frac{1}{(k-0.3)^2 + 0.04^2} + \frac{1}{(k-0.8)^2 + 0.06^2},$$

evaluated at K equidistant samples in $[0,1]$. We take $N = 2$, $I_1 = 10$, $I_2 = 25$, $K = I_1 I_2 = 250$, and $M = 350$. The NLS algorithm with random initialization is used for $R = \{1, 2, 3\}$ to compute a solution. In Figure 3.3, one can see that the accuracy of the approximation increases when using higher-rank values.

3.4.2 Comparison of methods

We compare the algebraic method in Algorithm 3.1, the NLS method in Algorithm 3.2, and the naive method, i.e., the trivial case of the algebraic method, see subsection 3.3.1. Remember that the latter can be computed by first solving the unstructured system and afterwards fitting the CPD structure on the obtained solution. Consider an LS-CPD with $N = 3$, $I_1 = I_2 = I_3 = I = 3$, $R = 1$, $K = I^3 = 27$. We choose $M^{(\text{min})} \leq M \leq K$ with $M^{(\text{min})} = 8$ which equals the minimal value of M to obtain a (generically) unique solution according to Lemma 1. We report the median relative error on the solution $\epsilon_{\mathbf{x}}$ and the computation time across 100 experiments in Figure 3.4. Timing experiments are performed on a standard laptop (quad core i7-4810MQ @ 2.80 GHz, 16 GB RAM, PM851 SSD) running Matlab 2016a on Windows 10. The naive method fails when $M < K$ because we solve an underdetermined linear system, resulting in a nonunique solution due to the nonemptiness of the null space of $\mathbf{A}$. The algebraic method works well, but fails if $M \leq 10$ because then the dimension of the null space of $\tilde{\mathbf{A}}$ in (3.10) is larger than one, see subsection 3.3.1. For $M = K$, the algebraic method coincides with the

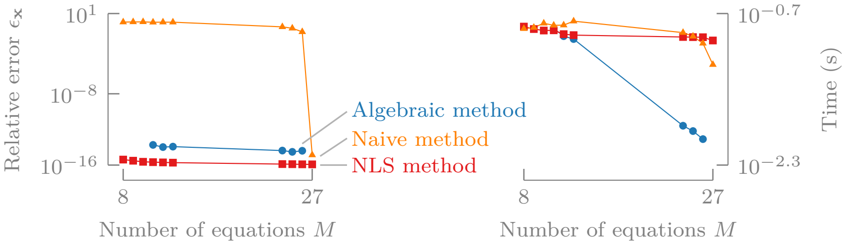


Figure 3.4: The naive method fails for an underdetermined LS-CPD while the NLS and algebraic method both perform well. The computational complexity of the algebraic method is much higher than the other two methods, especially for the highly underdetermined case (i.e., M close to the number of free variables).

naive method. The NLS method performs well for all M using five random initializations. Note that NLS typically needs many random initializations when M is close to $M^{(\min)}$. The accuracy is slightly higher than the algebraic method. The computational cost of the algebraic method increases when M decreases because $\tilde{\mathbf{A}}$ in (3.10) depends quadratically on L which is the dimension of the null space of $\mathbf{A}$.

3.4.3 Initialization methods

The algebraic method in Algorithm 3.1 finds the exact solution in the case of exact problems. In the case of perturbed problems, however, the solution can be used to obtain a good initialization for optimization-based methods such as the NLS algorithm from subsection 3.3.2. Often the algebraic solution provides a better starting value for optimization-based algorithms than a random initialization. We illustrate this for an underdetermined LS-CPD of the form (3.2) with $N = 3$, $R = 1$, $I_1 = I_2 = I_3 = I = 4$, $K = I^3 = 64$, and $M = 60$. We compute a solution using the NLS algorithm with random and algebraic initialization. In Figure 3.5, we report the median number of iterations across 20 experiments for several values of the SNR (right); we also show the convergence plot for 20 dB SNR on the left. By starting the NLS algorithm from the algebraic solution instead of using a random initialization, we need fewer iterations to achieve convergence. Importantly, the algebraic method can still find a solution in the noisy case but the accuracy is typically low. Optimization-based methods such as the NLS algorithm can use this solution as an initialization and improve the accuracy.

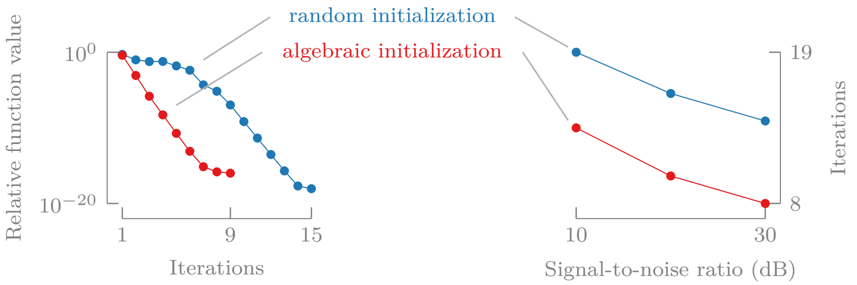


Figure 3.5: By initializing the NLS algorithm with the algebraic solution instead of using a random initialization, fewer iterations are needed to achieve convergence. *The relative function value is defined as the difference in objective function value between every two succesive iterations, relative to its initial value.*

3.4.4 Preconditioner

The overall computation time of the NLS algorithm can be reduced by reducing the computational cost per iteration or the number of iterations. Remember that we solve the normal equations in the NLS algorithm inexactly via a number of PCG iterations. Good preconditioning is essential to lower the number of CG iterations and, consequently, reduce the per-iteration complexity of the NLS algorithm. Here, we compare the Jacobi and block-Jacobi preconditioner (PC), see section 3.3.

Consider an LS-CPD problem with $N = 3$, $R = 3$, $I_1 = 250$, $I_2 = I_3 = 10$, $K = I_1 I_2 I_3 = 25000$. We consider three different scenarios: the highly underdetermined case ($M = R(I_1 + I_2 + I_3) + 5 = 815$), the underdetermined case ($M = 1.5R(I_1 + I_2 + I_3) = 1215$), and the square case $M = K = 25000$. We simulate a typical iteration of the NLS algorithm as follows. We compute the Gramian $\mathbf{H}$ and the gradient $\mathbf{g}$ for random factor matrices $\mathbf{U}^{(n)}$ and solve the normal equations in (3.16) using PCG until convergence (i.e., up to a relative error on the residual of 10^{-6}). In Table 3.2 we report the average number of CG iterations across 50 experiments when using no PC, the Jacobi PC, and the block-Jacobi PC for the three different scenarios. In this experiment, the block-Jacobi preconditioner reduces the number of CG iterations more than the Jacobi preconditioner, especially for the highly underdetermined case. In the square case, both PCs have similar performance, but the Jacobi PC is preferred because of its lower computational complexity. We have observed that the overall computation time of the algorithm can be reduced by using a preconditioner.

Table 3.2: Both PCs reduce the number of CG iterations in the underdetermined and square case. In the highly underdetermined case only the block-Jacobi PC can reduce the number of CG iterations. *We reported the average (and standard deviation of the) number of CG iterations across 50 experiments*

Scenario	No PC	Jacobi PC	block-Jacobi PC
highly underdetermined	810 (0)	790 (30)	644 (55)
underdetermined	181 (38)	56 (2)	46 (2)
square	45 (12)	12 (2)	12 (2)

3.5 Applications

LS-CPDs provide a generic framework that can be used in a wide range of applications. In this chapter, we illustrate with three applications in classification, multilinear algebra and signal processing, respectively. First, LS-CPDs are used for tensor-based face recognition in subsection 3.5.1. The technique is very generic and can be used for other classification tasks as well. For example, a similar method was used in [23] for irregular heartbeat classification in the analysis of electrocardiogram data. Next, it is shown in subsection 3.5.2 that tensors that have particular multilinear singular values can be constructed using LS-CPDs. Finally, in subsection 3.5.3, LS-CPDs are used for the blind deconvolution of CM signals.

3.5.1 Tensor-based face recognition using LS-CPDs

LS-CPDs can be used for generic classification tasks which is illustrated here using tensor-based face recognition [25], [204]. Consider a set of matrices of size $M_x \times M_y$ representing the facial images of J persons, taken under I different illumination conditions. All *vectorized* images of length $M = M_x M_y$ are stacked in a third-order tensor $\mathcal{D} \in \mathbb{K}^{M \times I \times J}$ with modes pixels (px) × illumination (i) × persons (p). Next, we perform a multilinear analysis by computing a (truncated) MLSVD of the tensor $\mathcal{D}$, i.e., we have:

$$\mathcal{D} \approx \mathcal{S} \cdot_1 \mathbf{U}_{\mathrm{px}} \cdot_2 \mathbf{U}_{\mathrm{i}} \cdot_3 \mathbf{U}_{\mathrm{p}}.$$

with $\mathbf{U}_{\mathrm{px}}$, $\mathbf{U}_{\mathrm{i}}$, and $\mathbf{U}_{\mathrm{p}}$ forming an orthonormal basis for the pixel, illumination, and person mode, respectively. The core tensor $\mathcal{S}$ explains the interaction between the different modes. The vectorized image $\mathbf{d} \in \mathbb{K}^{M}$ for a *particular* illumination i and person p satisfies:

$$\mathbf{d} = (\mathcal{S} \cdot_1 \mathbf{U}_{\mathrm{px}}) \cdot_2 \mathbf{c}_{\mathrm{i}}^{\mathrm{T}} \cdot_3 \mathbf{c}_{\mathrm{p}}^{\mathrm{T}} \tag{3.22}$$

with $\mathbf{c}_{\mathrm{i}}^{\mathrm{T}}$ and $\mathbf{c}_{\mathrm{p}}^{\mathrm{T}}$ rows of $\mathbf{U}_{\mathrm{i}}$ and $\mathbf{U}_{\mathrm{p}}$, with $\mathbf{U}_{\mathrm{p}}$ acting as a database. The mode-1 unfolding of (3.22) is an LS-CPD of the form (3.4):

$$\mathbf{d} = \mathbf{U}_{\mathrm{px}}\mathbf{S}_{(1)}(\mathbf{c}_{\mathrm{p}} \otimes \mathbf{c}_{\mathrm{i}}).$$

Consider a previously unknown image $\mathbf{d}^{(\mathrm{new})}$ of a person that is included in the database. Classification or recognition of this person corresponds to finding the coefficient vector $\mathbf{c}_{\mathrm{p}}$, i.e., we solve an LS-CPD of the form:

$$\mathbf{d}^{(\mathrm{new})} = \mathbf{U}_{\mathrm{px}}\mathbf{S}_{(1)}\left(\mathbf{c}_{\mathrm{p}}^{(\mathrm{new})} \otimes \mathbf{c}_{\mathrm{i}}^{(\mathrm{new})}\right),$$

resulting into estimates $\tilde{\mathbf{c}}_{\mathrm{p}}^{(\mathrm{new})}$ and $\tilde{\mathbf{c}}_{\mathrm{i}}^{(\mathrm{new})}$. The coefficient vector for the person dimension $\tilde{\mathbf{c}}_{\mathrm{p}}^{(\mathrm{new})}$ is compared with the rows of $\mathbf{U}_{\mathrm{p}}$ using the Frobenius norm of the difference (after fixing scaling and sign invariance)[1]. We then classify the person in the image according to the label of the closest match.

Let us illustrate the above strategy for the extended YaleB dataset[2]. This real-life dataset consists of cropped facial images of 39 persons in 64 illumination conditions. We remove illumination conditions for which some of the images are missing and retain one of the conditions as test data, resulting into $I = 56$ conditions. We vectorize each image of 51×58 pixels into a vector of length $M = 2958$ for $J = 37$ persons. The resulting data tensor $\mathcal{D}$ has size $2958 \times 56 \times 37$. We compute the MLSVD of $\mathcal{D}$ using a randomized algorithm called `mlsvd_rsi`, which is faster than non-randomized algorithms but achieves similar accuracy [211]. We compress the pixel mode to reduce noise influences. As such, we obtain a core tensor $\mathcal{S} \in \mathbb{K}^{500\times 56\times 37}$ and matrices $\mathbf{U}_{\mathrm{px}} \in \mathbb{K}^{2958\times 500}$, $\mathbf{U}_{\mathrm{i}} \in \mathbb{K}^{56\times 56}$, and $\mathbf{U}_{\mathrm{p}} \in \mathbb{K}^{37\times 37}$. We use the NLS algorithm to compute a solution, starting from a random initialization. We project the new image $\mathbf{d}^{(\mathrm{new})}$ onto the column space of the pixel matrix $\mathbf{U}_{\mathrm{px}}$ in order to decrease computation time, i.e., $\mathbf{b} = \mathbf{U}_{\mathrm{px}}^{\mathrm{T}}\mathbf{d}^{(\mathrm{new})}$. We compare the estimated coefficient vector with $\mathbf{U} = \mathbf{U}_{\mathrm{p}}$. To accommodate for scaling and sign invariance, we normalize the rows of $\mathbf{U}$ and $\tilde{\mathbf{c}}_{\mathrm{p}}^{(\mathrm{new})}$ as follows: a vector $\mathbf{c}$ is normalized as $\mathrm{sign}(c_1)\frac{\mathbf{c}}{||\mathbf{c}||}$. On the left in Figure 3.6, we see the facial image of a person that is known to our model but for a new illumination condition. In the middle one can see the reconstruction of the image using the estimated coefficient vectors. Moreover, we correctly classified the person as the person on the right in Figure 3.6.

---

[1] Note that other metrics can be used such as cosine similarity [148].

[2] Available from `http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html`.

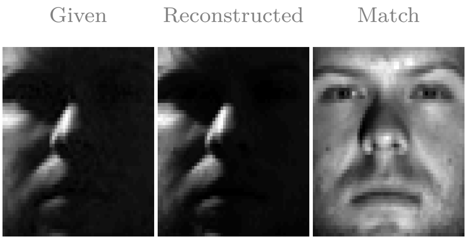


Figure 3.6: Correct classification of a facial image under a *new* illumination condition.

3.5.2 Constructing a tensor that has particular multilinear singular values

Constructing a matrix with particular singular values is trivial. One can simply use the SVD: $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$ in which $\mathbf{\Sigma}$ is a diagonal matrix containing the given singular values and $\mathbf{U}$ and $\mathbf{V}$ are random orthogonal matrices. For tensors, this is not straightforward. It is of fundamental importance to understand the behavior of multilinear singular values [77], [100], [101]. In this section, we show how one can construct an all-orthogonal tensor $\mathcal{T} \in \mathbb{R}^{I\times J\times K}$ with particular multilinear singular values using an LS-CPD. Consider the following expressions:

$$\begin{cases} \mathbf{T}_{(1)}\mathbf{T}_{(1)}^{\mathrm{T}} = \mathbf{\Sigma}^{(1)}, \\ \mathbf{T}_{(2)}\mathbf{T}_{(2)}^{\mathrm{T}} = \mathbf{\Sigma}^{(2)}, \\ \mathbf{T}_{(3)}\mathbf{T}_{(3)}^{\mathrm{T}} = \mathbf{\Sigma}^{(3)}, \end{cases} \tag{3.23}$$

in which $\mathbf{\Sigma}^{(n)} = \mathrm{diag}(\boldsymbol{\sigma}^{(n)^2})$ is a diagonal matrix containing the squared multilinear singular values $\boldsymbol{\sigma}^{(n)}$, $n = 1, 2, 3$. Expression (3.23) states that $\mathcal{T}$ is all-orthogonal and has multilinear singular values $\boldsymbol{\sigma}^{(n)}$. In order to reformulate (3.23) as an LS-CPD, we only take the upper triangular parts into account because of symmetry in the left- and right-hand side in (3.23), leading to the following equations for the first expression in (3.23):

$$\sum_{j,k} t_{ijk}t_{ijk} = \left(\sigma_i^{(1)}\right)^2, \text{ for } 1 \leq i \leq I,$$

$$\sum_{j,k} t_{i_1jk}t_{i_2jk} = 0, \text{ for } 1 \leq i_1 < i_2 \leq I,$$

and similarly for the second and third expression. We can write this more compactly as an LS-CPD:

$$\mathbf{A}(\mathbf{u} \otimes \mathbf{u}) = \mathbf{b} \quad \text{with} \quad \mathbf{u} = \mathrm{vec}(\mathcal{T}).$$

Table 3.3: The LS-CPD method for constructing a tensor with particular multilinear singular values is faster than APM. *This is illustrated by comparing the median computation time (in seconds) across 20 experiments for an $I_1 \times I_2 \times I_3$ tensor with $I_1 = I_2 = 10\alpha$ and $I_3 = 5\alpha$ in which $\alpha = \{1, 5, 10\}$.*

	$\alpha = 1$	$\alpha = 5$	$\alpha = 10$
Alternating projection method (APM) [100]	0.100	34.4	1747
Construction of **A**	0.004	1.4	23
Initialization (i.e., one iteration of APM)	0.002	0.4	12
LS-CPD	0.023	15.4	444
Total computation time of LS-CPD	0.029	17.2	479

$\mathbf{A}$ is a binary and sparse matrix of size $I_{\mathrm{A}} \times J_{\mathrm{A}}^2$ with $I_{\mathrm{A}} = \sum_{n=1}^{N} \frac{I_n(I_n+1)}{2}$ and $J_{\mathrm{A}} = \prod_{n=1}^{N} I_n$. The right-hand side $\mathbf{b} \in \mathbb{K}^{I_{\mathrm{A}}}$ is defined as

$$\mathbf{b} = \left[\operatorname{triu}\left(\mathbf{\Sigma}^{(1)}\right); \operatorname{triu}\left(\mathbf{\Sigma}^{(2)}\right); \cdots; \operatorname{triu}\left(\mathbf{\Sigma}^{(N)}\right)\right]$$

in which each entry is either zero or a squared multilinear singular value. One can show that the Jacobian for this particular problem is also a sparse matrix of size $I_{\mathrm{A}} \times J_{\mathrm{A}}$ with $\sum_{n=1}^{N} I_n$ nonzeros in each column. More specifically, the Jacobian has the form: $\mathbf{J} = \mathbf{J}^{(1)} + \mathbf{J}^{(2)}$ with $\mathbf{J}^{(1)}$ and $\mathbf{J}^{(2)}$ the derivative to the first and second $\mathbf{u}$, respectively. Computing the sub-Jacobians $\mathbf{J}^{(n)}$ is reduced to filling in elements of $\mathcal{T}$ at the correct position in $\mathbf{J}^{(n)}$ for the orthogonality constraints. For the multilinear singular value constraints, one has to multiply by two. By exploiting the structure, no additional operations are required. We implemented this using a C/mex function that replaces entries to avoid the overhead of constructing sparse matrices in MATLAB. The Gramian of the Jacobian is computed using sparse matrix-vector products.

We compare the optimized NLS algorithm with the alternating projection method (APM) [100] in terms of computation time needed to construct an $I_1 \times I_2 \times I_3$ tensor with given multilinear singular values. We take $I_1 = I_2 = 10\alpha$ and $I_3 = 5\alpha$ in which $\alpha = \{1, 5, 10\}$. The multilinear singular values are chosen by constructing a tensor that can be written as a sum of a multilinear rank-$(L_1, 1, L_1)$ term and a multilinear rank-$(1, L_2, L_2)$ term with $L_1 = I_3 - 1$ and $L_2 = I_3 + 1$. The elements of the factor matrices are drawn from the standard normal distribution. We normalize the multilinear singular values such that the tensor has unit Frobenius norm. We initialize the NLS algorithm with the solution obtained after one iteration of APM. In Table 3.3, we report the median computation time across 20 experiments. The time to construct $\mathbf{A}$ is reported separately because it depends only on the size of the tensor and its construction has to be performed only once. Clearly, the computation time of LS-CPD is much lower than APM, even if we include the time needed to construct $\mathbf{A}$.

3.5.3 Blind deconvolution of constant modulus signals

LS-CPDs can also be used in signal processing applications. We illustrate this by reformulating the blind deconvolution of a CM signal [206] as the computation of an LS-CPD. In this chapter, we investigate the single-input-single-output (SISO) case using an autoregressive (AR) model [137], i.e., we have:

$$\sum_{l=0}^{L} w_l \cdot y[k-l] = s[k] + n[k], \text{ for } 1 \leq k \leq K, \tag{3.24}$$

with $y[k]$, $s[k]$, and $n[k]$ being the measured output, the input, and the additive noise at the kth time instance, respectively. The lth filter coefficient is denoted as w_l. Assume we have $K+L-1$ samples $y[-L+1], \ldots, y[K]$ and let $\mathbf{Y} \in \mathbb{K}^{L \times K}$ be a Toeplitz matrix defined as $y_{lk} = y[k-l]$. Also, the filter coefficients are collected in $\mathbf{w} \in \mathbb{K}^{L}$ and the source vector $\mathbf{s} \in \mathbb{K}^{K}$ is defined as $s_k = s[k]$. We ignore the noise in the derivation of our method for simplicity. Equation (3.24) can then be expressed in matrix form as:

$$\mathbf{Y}^{\mathrm{T}} \mathbf{w} = \mathbf{s}. \tag{3.25}$$

The goal of blind deconvolution is to find the vector $\mathbf{w}$ using only the measured output values [1]. In order to make this problem identifiable, additional prior knowledge has to be exploited. Here, we assume that the input signal has constant modulus, i.e., each sample s_k satisfies the following property [64]:

$$|s_k|^2 = s_k \cdot \overline{s}_k = c, \text{ for } 1 \leq k \leq K \tag{3.26}$$

with c being the squared constant modulus which is known *a priori*. By using (3.25) in (3.26), we obtain:

$$\left(\mathbf{y}_k^{\mathrm{T}} \mathbf{w}\right) \left(\overline{\mathbf{y}_k^{\mathrm{T}} \mathbf{w}}\right) = c, \quad \text{or, equivalently,} \quad \left(\mathbf{y}_k \otimes \overline{\mathbf{y}}_k\right)^{\mathrm{T}} \left(\mathbf{w} \otimes \overline{\mathbf{w}}\right) = c, \tag{3.27}$$

in which $\mathbf{y}_k$ is the kth column of $\mathbf{Y}$, for $1 \leq k \leq K$. Taking into account all equations, (3.27) reduces to the following LS-CPD:

$$\left(\mathbf{Y} \odot \overline{\mathbf{Y}}\right)^{\mathrm{T}} \left(\mathbf{w} \otimes \overline{\mathbf{w}}\right) = c \cdot \mathbf{1}_K. \tag{3.28}$$

We illustrate the approach by means of the following straightforward example. Consider an AR model of degree $L = 5$ with uniformly distributed coefficients between zero and one, sample length $K = 100$, and $c = 1$. We perturb the measurements with additive Gaussian noise which is scaled to obtain a particular SNR. We solve (3.28) by relaxing $\overline{\mathbf{w}}$ to $\mathbf{v}$:

$$\left(\mathbf{Y} \odot \overline{\mathbf{Y}}\right)^{\mathrm{T}} \left(\mathbf{w} \otimes \mathbf{v}\right) = c \cdot \mathbf{1}_K \tag{3.29}$$

using the NLS algorithm with the Jacobi PC and starting from the algebraic

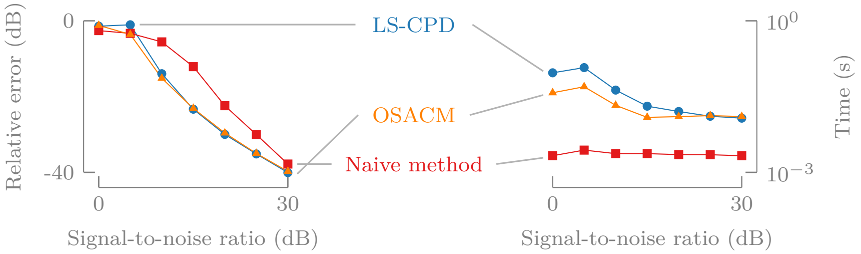


Figure 3.7: The LS-CPD approach obtains more accurate results than the naive method and achieves similar accuracy as the dedicated OSACM method. The run-time of LS-CPD is slightly higher than OSACM for this example. *The naive method has a lower run-time than the other methods.*

solution. In Figure 3.7, we report the median relative error on $\mathbf{w}$ and the median run-time across 50 experiments for several values of the SNR. Timing experiments are performed on a standard laptop (quad core i7-4810MQ @ 2.80 GHz, 16 GB RAM, PM851 SSD) running Matlab 2016a on Windows 10. We compare our approach to the naive method, i.e., the method that relaxes the Kronecker structure in (3.29), solves the system, and subsequently fits the Kronecker structure to the least-squares solution. These are the core elements of the well-known analytical constant modulus algorithm (ACMA) [206], [207]. We also compare with a state-of-the-art SISO CM algorithm (CMA) called optimal step-size CMA (OSCMA) [191], [219]. It is clear that our *generic* LS-CPD method obtains more accurate results than the relaxation-based technique and achieves similar accuracy as the *dedicated* OSCMA method. Also, the run-time of the LS-CPD approach is only slightly higher than the OSCMA method in this example, but can be further reduced by exploiting the structure in the coefficient matrix.

3.6 Conclusion and future research

We presented a new framework for linear systems with a CPD constrained solution (LS-CPD). We defined the LS-CPD problem, discussed links between particular types of structured coefficient matrices and the CPD problem, and derived a condition guaranteeing generic uniqueness of the solution. In contrast to the naive method, the proposed algebraic and optimization-based methods allow one to solve the LS-CPD problem in the underdetermined case. Although we focused on the Gauss–Newton (GN) method, the derivations of the expressions for the objective function, gradient, Gramian, and Gramian-vector product can also be used to implement various nonlinear least-squares and quasi-Newton algorithms. Numerical experiments show that the algebraic method is a good starting point for optimization-based methods. We also compared the effectiveness of two preconditioners for the GN method. The wide applicability of LS-CPDs is illustrated with three applications from classification, multilinear algebra, and signal processing. Importantly, we have explained that many classification tasks can be formulated as the computation of an LS-CPD. In order to reduce the per-iteration computational complexity of the NLS algorithm, application-dependent structure can be exploited, as we have shown with the construction of tensors that have particular singular values.

NLS algorithm for Kronecker-structured linear systems with a CPD constrained solution

4

ABSTRACT | In various applications within signal processing, system identification, pattern recognition, and scientific computing, the canonical polyadic decomposition (CPD) of a higher-order tensor is only known via general linear measurements. In this chapter, we show that the computation of such a CPD can be reformulated as a sum of CPDs with linearly constrained factor matrices by assuming that the measurement matrix can be approximated by a sum of a (small) number of Kronecker products. By properly exploiting the hypothesized structure, we can derive an efficient non-linear least-squares algorithm, allowing us to tackle large-scale problems.

This chapter is a slightly adapted version of M. Boussé, N. Sidiropoulos, and L. De Lathauwer, “NLS algorithm for Kronecker-structured linear systems with a CPD constrained solution”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-13, 2019, Accepted for publication in the proceedings of the 27th European Signal Processing Conference (EUSIPCO 2019, A Coruña, Spain).

4.1 Introduction

Even though the decomposition of a tensor that is known *explicitly* is a prevalent problem in signal processing and machine learning [41], [168], we often want to compute a decomposition of a tensor that is only known via linear measurements [26]. Applications can be found in a wide range of domains such as signal processing [26], [221], system identification [17], [143], pattern recognition [23], [25], [26], [97], and scientific computing [8], [14], [83], [96]. By limiting ourselves to a canonical polyadic decomposition (CPD) in this chapter, we can formulate the problem as a linear system of equations with a CPD constrained solution (LS-CPD) [26], i.e., $\mathbf{A}\mathbf{x} = \mathbf{b}$ with $\mathbf{x} = \operatorname{vec}(\text{CPD})$. Or, equivalently, we want to compute a CPD of a tensor $\mathcal{X} = \operatorname{unvec}(\mathbf{x})$ that is only defined *implicitly* via the solution of a linear system.

By fully exploiting all structure of the measurement matrix $\mathbf{A}$, the computational complexity of a dedicated algorithm can be significantly reduced, enabling efficient processing for large-scale problems [26]. For example, if $\mathbf{A}$ is equal to the identity (or a diagonal) matrix, the problem reduces to a (weighted) CPD of a known tensor, allowing efficient computations [16], [154], [180]. In the special case where $\mathbf{A}$ is sparse, we can also obtain efficient algorithms, see [26].

In this chapter, we assume that $\mathbf{A}$ can be written as a sum of L Kronecker products. This strategy is employed to reduce the computational complexity of algorithms in various applications within signal processing [20], [21], system identification [17], [143], [173], [174], and tensor-based scientific computing [8], [14], [96], among others. Depending on the application, the products are considered to be given or they can be computed. As a matter of fact, *any* measurement matrix can be approximated by a sum of Kronecker products for sufficiently large L. For a given L, a least-squares approximation can be computed via a Kronecker product decomposition [9], [147], [198].

By explicitly leveraging the Kronecker structure of $\mathbf{A}$, the LS-CPD problem can be reformulated as a sum of L CPDs with linear constraints. In constrast to existing methods that employ projection [8], alternating least-squares [14], [221], or a gradient approach [83], we develop numerical optimization-based techniques such as quasi-Newton (qN) and nonlinear least-squares (NLS) with known convergence properties [150]. By carefully exploiting all available structure, our algorithm can tackle large-scale problems [210]. For $L = 1$, the problem can be related to CANDELINC [35], [125], which can be computed efficiently in the dense [35] and sparse [212] case.

In the remainder of this section, we discuss notations and basic definitions. In section 4.2, we reformulate the LS-CPD via Kronecker structure. By properly exploiting the structure, we obtain an efficient optimization-based algorithm in section 4.3. Numerical experiments are discussed in section 4.4.

4.1.1 Notations and basic definitions

A tensor (denoted by $\mathcal{A}$) is a higher-order generalization of a vector and a matrix (denoted by $\mathbf{a}$ and $\mathbf{A}$, respectively). A mode-n vector of a tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ (with $\mathbb{K}$ meaning $\mathbb{R}$ or $\mathbb{C}$) is defined by fixing every index except the nth. The mode-n unfolding of $\mathcal{A}$ is the matrix $\mathbf{A}_{(n)}$ with the mode-n vectors as its columns (using the ordering convention in [125]). The vectorization of $\mathcal{A}$, denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \dots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1 + \sum_{k=1}^{N}(i_k - 1)J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$ (with $\prod_{m}^{k-1}(\cdot) = 1$ if $m > k-1$). The $\operatorname{unvec}(\cdot)$ operation is defined as the inverse of $\operatorname{vec}(\cdot)$. We denote the outer, Kronecker and Khatri–Rao product as $\otimes$, $\otimes$ and $\odot$, respectively. We say that a Nth-order tensor has rank one if it can be written as the outer product of N nonzero vectors. The rank of a tensor is defined as the minimal number of rank-1 terms that generate the tensor as their sum.

4.1.2 Canonical Polyadic Decomposition (CPD)

The CPD is an important tool for tensor analysis in signal processing, data mining and machine learning [41], [125], [168]. The decomposition is unique under rather mild conditions [72], [75], which is a powerful advantage over matrices [168].

Definition 23. A *polyadic decomposition* (PD) writes an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ as a sum of R rank-1 terms:

$$\mathcal{A} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(N)} = \left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \dots, \mathbf{U}^{(N)} \right]\!\right].$$

The columns of the factor matrices $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$ are equal to the factor vectors $\mathbf{u}_r^{(n)}$ for $1 \leq r \leq R$. The PD is said to be *canonical* (CPD) when R is equal to the rank of $\mathcal{A}$.

The CANDELINC model is a popular tool to incorporate prior knowledge in the CPD by means of linear constraints, allowing one to improve the accuracy and/or interpretability [35], [212]. One assumes that $\mathbf{U}^{(n)} = \mathbf{A}^{(n)}\mathbf{C}^{(n)}$ in which $\mathbf{A}^{(n)}$ is known and $\mathbf{C}^{(n)}$ is the unknown coefficient matrix.

4.1.3 Identities and derivatives

We use the following identities in the derivation of our algorithm [136]:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}, \tag{4.1}$$

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \odot \mathbf{D}) = \mathbf{AC} \odot \mathbf{BD}, \tag{4.2}$$

$$(\mathbf{A} \odot \mathbf{B})^{\mathrm{T}} (\mathbf{C} \odot \mathbf{D}) = \mathbf{A}^{\mathrm{T}}\mathbf{C} * \mathbf{B}^{\mathrm{T}}\mathbf{D}, \tag{4.3}$$

$$\operatorname{vec}(\mathbf{ABC}) = \left(\mathbf{C}^{\mathrm{T}} \otimes \mathbf{A}\right) \operatorname{vec}(\mathbf{B}), \tag{4.4}$$
$$\operatorname{vec}\left([\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]_R\right) = \left(\mathbf{C} \odot \mathbf{B} \odot \mathbf{A}\right) \mathbf{1}_R. \tag{4.5}$$

Given a matrix $\mathbf{P}^{(n)}$ that permutes the nth mode of a vectorized tensor to the first mode and ${\mathbf{P}^{(n)}}^{\mathrm{T}}\mathbf{P}^{(n)} = \mathbf{I}$, we have:

$$\begin{aligned}
&\mathbf{P}^{(n)}\operatorname{vec}\left(\left[\!\!\left[\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(N)}\right]\!\!\right]\right) = \\
&\operatorname{vec}\left(\left[\!\!\left[\mathbf{U}^{(n)}, \mathbf{U}^{(1)}, \dots, \mathbf{U}^{(n-1)}, \mathbf{U}^{(n+1)}, \dots, \mathbf{U}^{(N)}\right]\!\!\right]\right).
\end{aligned}$$

By defining $\mathbf{V}^{\{n\}} = \odot_{q=1, q\neq N-n+1}^{N} \mathbf{U}^{(N-q+1)}$, we obtain:

$$\begin{aligned}
&{\mathbf{P}^{(n)}}^{\mathrm{T}}\left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)\operatorname{vec}(\mathbf{X}) = \\
&\quad\operatorname{vec}\left(\left[\!\!\left[\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(n-1)}, \mathbf{X}, \mathbf{U}^{(n+1)}, \dots, \mathbf{U}^{(N)}\right]\!\!\right]\right), \\
&{\mathbf{P}^{(n)}}^{\mathrm{T}}\operatorname{vec}\left(\mathbf{U}^{(n)}{\mathbf{V}^{\{n\}}}^{\mathrm{T}}\right) = \operatorname{vec}\left(\left[\!\!\left[\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(n)}\right]\!\!\right]\right).
\end{aligned} \tag{4.6}$$

Finally, we also use the following derivative [124]:

$$\frac{\partial \operatorname{vec}\left([\![\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(n)}]\!]\right)}{\partial \operatorname{vec}\left(\mathbf{U}^{(n)}\right)} = {\mathbf{P}^{(n)}}^{\mathrm{T}}\left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right). \tag{4.7}$$

4.2 Kronecker-structured LS-CPD as a sum of CPDs with linearly constrained factor matrices

We consider a CPD of a tensor that is only known via linear measurements. This can be formulated as a linear system with a CPD constrained solution (LS-CPD) [26]:

$$\mathbf{Ax} = \mathbf{b} \quad \text{with} \quad \mathbf{x} = \operatorname{vec}\left(\left[\!\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \dots, \mathbf{U}^{(N)}\right]\!\!\right]\right) \tag{4.8}$$

in which $\mathbf{A} \in \mathbb{K}^{M\times K}$, $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$, and $K = \prod_{n=1}^{N} I_n$. Additionally, we assume that $\mathbf{A}$ admits, or can be well approximated by, a sum of L Kronecker products of smaller matrices $\mathbf{A}^{(n,l)} \in \mathbb{K}^{J_n \times I_n}$, for $1 \leq n \leq N$, and $M = \prod_{n=1}^{N} J_n$, i.e.,

$$\mathbf{A} = \sum_{l=1}^{L} \mathbf{A}^{(N,l)} \otimes \mathbf{A}^{(N-1,l)} \otimes \dots \otimes \mathbf{A}^{(1,l)}. \tag{4.9}$$

By assuming Kronecker structure (4.9), the LS-CPD problem in (4.8) can be reduced to a sum of CPDs with linear constraints. First, combine (4.8) and (4.9) to obtain:

$$\sum_{l=1}^{L}\left(\mathbf{A}^{(N,l)}\otimes\cdots\otimes\mathbf{A}^{(1,l)}\right)\left(\mathbf{U}^{(N)}\odot\cdots\odot\mathbf{U}^{(1)}\right)\cdot\mathbf{1}_R=\mathbf{b}.$$

By using the mixed-product rule in (4.2), we can write that:

$$\sum_{l=1}^{L}\left(\mathbf{A}^{(N,l)}\mathbf{U}^{(N)}\odot\cdots\odot\mathbf{A}^{(1,l)}\mathbf{U}^{(1)}\right)\cdot\mathbf{1}_R=\mathbf{b}. \tag{4.10}$$

By defining factor matrices $\mathbf{V}^{(n,l)}=\mathbf{A}^{(n,l)}\mathbf{U}^{(n)}$, for $1\leq n\leq N$, we can write (4.10) as a sum of L CPDs with linearly constrained factor matrices of a tensor $\mathcal{B}=\text{unvec}(\mathbf{b})$ as follows:

$$\mathcal{B}=\sum_{l=1}^{L}\left[\!\left[\mathbf{V}^{(1,l)},\mathbf{V}^{(2,l)},\ldots,\mathbf{V}^{(N,l)}\right]\!\right]. \tag{4.11}$$

If $L=1$, it is clear that (4.11) can be related to the well-known CANDELINC. For $L>1$, we obtain a more general model.

By stacking the factor matrices $\mathbf{V}^{(n,l)}$, $1\leq l\leq L$, in a matrix $\mathbf{V}^{(n)}=\begin{bmatrix}\mathbf{V}^{(n,1)} & \mathbf{V}^{(n,2)} & \cdots & \mathbf{V}^{(n,L)}\end{bmatrix}\in\mathbb{K}^{J_n\times RL}$, (4.11) reduces to a rank-RL CPD with linear block constraints:

$$\mathcal{B}=\left[\!\left[\mathbf{V}^{(1)},\mathbf{V}^{(2)},\ldots,\mathbf{V}^{(N)}\right]\!\right].$$

If all $\mathbf{A}^{(n,l)}$ have full column rank, existing uniqueness results can be used, see [72], [75]. Depending on the application, we are interested in either interpretable, and therefore unique, factor matrices or a compact representation of the underlying tensor using factor matrices which do not need to be unique.

4.3 Nonlinear least-squares (NLS) algorithm

By properly exploiting the Kronecker structure, we derive an efficient NLS algorithm for (4.8)-(4.9). The computation can be formulated as an optimization problem as follows:

$$\min_{\mathbf{z}} f=\frac{1}{2}\,||\mathcal{F}||_{\text{F}}^{2}\quad\text{with }\mathcal{F}\text{ defined as in (4.13)}, \tag{4.12}$$

in which the variables $\mathbf{U}^{(n)}$, for $1\leq n\leq N$, are concatenated in a vector $\mathbf{z}\in\mathbb{K}^{RI^{+}}$ with $I^{+}=\sum_{n=1}^{N}I_n$, as follows: $\mathbf{z}=\left[\text{vec}\left(\mathbf{U}^{(1)}\right);\ \cdots\ ;\ \text{vec}\left(\mathbf{U}^{(N)}\right)\right]$.

The residual $\mathcal{F}$ is given by:

$$\mathcal{F} = \sum_{l=1}^{L} \left[\!\left[\mathbf{V}^{(1,l)}, \mathbf{V}^{(2,l)}, \ldots, \mathbf{V}^{(N,l)} \right]\!\right] - \mathcal{B} \tag{4.13}$$

with linear constraints $\mathbf{V}^{(n,l)} = \mathbf{A}^{(n,l)}\mathbf{U}^{(n)} \in \mathbb{K}^{J_n \times R}$.

We can solve the optimization problem in (4.12)-(4.13) using standard qN and NLS algorithms by deriving expressions for the evaluation of the objective function, gradient, Jacobian, Gramian, and Gramian-vector product. Importantly, we exploit all available structure in order to obtain efficient implementations. In this chapter, we focus on the Gauss–Newton (GN) algorithm [150], but the expressions can be used for other qN and NLS algorithms as well. In order to implement our algorithm, we use the complex optimization framework from [177], [179], [210], which provides qN and NLS implementations as well as line, plane search, and trust-region methods. Additionally, we provide a computational complexity analysis.

The GN method using dogleg trust-region solves (4.12) by linearizing the residual $\mathrm{vec}\,(\mathcal{F})$ in each iteration k and subsequently by solving the following least-squares problem [150]:

$$\min_{\mathbf{p}_k} \frac{1}{2} \left|\left|\mathrm{vec}\,(\mathcal{F}_k) + \mathbf{J}_k\mathbf{p}_k\right|\right|_{\mathrm{F}}^2 \quad \text{s.t.} \quad \left|\left|\mathbf{p}_k\right|\right| \leq \Delta_k \tag{4.14}$$

with step $\mathbf{p}_k = \mathbf{z}_{k+1} - \mathbf{z}_k$, Jacobian $\mathbf{J} = \mathrm{d}\,\mathrm{vec}\,(\mathcal{F})\,/\mathrm{d}\mathbf{z}$, and trust-region Δ_k. The exact solution to (4.14) is given by the linear system $\mathbf{H}_k\mathbf{p}_k = -\overline{\mathbf{g}}_k$ with $\mathbf{H}$ the Hessian, which we approximate with the Gramian of the Jacobian, and the conjugated gradient $\overline{\mathbf{g}} = (\partial f/\partial \mathbf{z})^{\mathrm{H}}$ [150]. The variables can then be updated as $\mathbf{z}_{k+1} = \mathbf{z}_k + \mathbf{p}_k$. We solve the linear system using several preconditioned conjugate gradient (CG) iterations in order to reduce computational complexity. The GN method is summarized in Algorithm 4.1.

Algorithm 4.1: Kronecker-structured linear system with a CPD constrained solution using Gauss–Newton with dogleg trust region.

```
Input: $\mathcal{B}$, $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$, and initial estimate for $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$
Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$
while not converged do
    Compute gradient $\mathbf{g}$ using (4.15) and (4.16).
    Use PCG to solve $\mathbf{Hp} = -\overline{\mathbf{g}}$ for $\mathbf{p}$ using Gramian-vector products in (4.17)-(4.18) and a block-Jacobi preconditioner as explained in subsection 3.3.2.
    Update $\mathbf{U}^{(n)}$, for $1 \leq n \leq N$, using dogleg trust region from $\mathbf{p}$, $\mathbf{g}$, and function evaluation (4.12).
end while
```

4.3.1 Objective function

The objective function f can be evaluated by taking the sum of squared entries of the residual $\mathcal{F}(\mathbf{z})$ as defined in (4.13).

4.3.2 Jacobian

The Jacobian $\mathbf{J}$ can be partitioned in the following way:

$$\mathbf{J} = \frac{\mathrm{d}\,\mathrm{vec}\left(\mathcal{F}\right)}{\mathrm{d}\mathbf{z}} = \begin{bmatrix}\mathbf{J}^{(1)} & \mathbf{J}^{(2)} & \cdots & \mathbf{J}^{(N)}\end{bmatrix} \in \mathbb{K}^{K \times RI^{+}}.$$

with the nth sub-Jacobian $\mathbf{J}^{(n)}$ defined as:

$$\mathbf{J}^{(n)} = \sum_{l=1}^{L} \mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{V}^{\{n,l\}} \otimes \mathbf{A}^{(n,l)}\right) \in \mathbb{K}^{K \times RI_n}$$

with $\mathbf{V}^{\{n,l\}} = \odot_{q=1, q\neq N-n+1}^{N} \mathbf{V}^{(n,l)}$.

Proof.

$$\begin{aligned}
\mathbf{J}^{(n)} = \frac{\partial \mathrm{vec}\left(\mathcal{F}\right)}{\partial \mathrm{vec}\left(\mathbf{U}^{(n)}\right)} &= \sum_{l=1}^{L} \frac{\partial \left(\left[\!\left[\mathbf{V}^{(1,l)}, \ldots, \mathbf{V}^{(N,l)}\right]\!\right]\right)}{\partial \mathrm{vec}\left(\mathbf{U}^{(n)}\right)} \\
&= \sum_{l=1}^{L} \frac{\partial \left(\left[\!\left[\mathbf{A}^{(1,l)}\mathbf{U}^{(1)}, \ldots, \mathbf{A}^{(N,l)}\mathbf{U}^{(N)}\right]\!\right]\right)}{\partial \mathrm{vec}\left(\mathbf{U}^{(n)}\right)} \\
&= \sum_{l=1}^{L} \left(\bigotimes_{n=N}^{1} \mathbf{A}^{(n,l)}\right) \cdot \frac{\partial \mathrm{vec}\left(\left[\!\left[\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N)}\right]\!\right]\right)}{\partial \mathrm{vec}\left(\mathbf{U}^{(n)}\right)} \\
&= \sum_{l=1}^{L} \left(\bigotimes_{n=N}^{1} \mathbf{A}^{(n,l)}\right) \mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \\
&= \sum_{l=1}^{L} \mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{V}^{\{n,l\}} \otimes \mathbf{A}^{(n,l)}\right).
\end{aligned}$$

Identities (4.2) and (4.5) enable the third equation. The last two equations are obtained by using (4.7) and (4.1)-(4.2), respectively.

4.3.3 Gradient

The gradient $\mathbf{g}$ can be partitioned in the following way

$$\mathbf{g} = \begin{bmatrix}\mathbf{g}^{(1)} & \mathbf{g}^{(2)} & \cdots & \mathbf{g}^{(N)}\end{bmatrix} \in \mathbb{K}^{RI^{+}}, \tag{4.15}$$

in which $\mathbf{g}^{(n)} \in \mathbb{K}^{RI_n}$ is defined by:

$$\mathbf{g}^{(n)} = \sum_{l=1}^{L} \operatorname{vec}\left(\mathbf{A}^{(n,l)^{\mathrm{T}}} \overline{\mathbf{F}}_{(n)} \mathbf{V}^{\{n,l\}}\right). \tag{4.16}$$

Proof. The nth subgradient is given by:

$$\begin{aligned}
\mathbf{g}^{(n)} &= \frac{\partial f}{\partial \operatorname{vec}\left(\mathbf{U}^{(n)}\right)} = \mathbf{J}^{(n)^{\mathrm{T}}} \overline{\operatorname{vec}\left(\mathcal{F}\right)} \\
&= \sum_{l=1}^{L} \left(\mathbf{V}^{\{n,l\}^{\mathrm{T}}} \otimes \mathbf{A}^{(n,l)^{\mathrm{T}}}\right) \mathbf{P}^{(n)} \overline{\operatorname{vec}\left(\mathcal{F}\right)} \\
&= \sum_{l=1}^{L} \operatorname{vec}\left(\mathbf{A}^{(n,l)^{\mathrm{T}}} \overline{\mathbf{F}}_{(n)} \mathbf{V}^{\{n,l\}}\right).
\end{aligned}$$

We use (4.4) to obtain the last equation, which can be computed efficiently using Tensorlab's `mtkrprod` implementation [215].

4.3.4 Gramian-vector product

We compute the product $\mathbf{J}^{(m)^{\mathrm{H}}} \mathbf{J}^{(n)} \operatorname{vec}\left(\mathbf{X}^{(n)}\right)$ efficiently, by first computing $\mathbf{t} = \mathbf{J}^{(n)} \operatorname{vec}\left(\mathbf{X}^{(n)}\right)$ with $\mathbf{X}^{(n)} \in \mathbb{K}^{I_n \times R}$:

$$\begin{aligned}
\mathbf{J}^{(n)} \operatorname{vec}\left(\mathbf{X}^{(n)}\right) &= \sum_{l=1}^{L} \mathbf{P}^{(n)^{\mathrm{T}}} \left(\tilde{\mathbf{V}}^{\{n,l\}} \otimes \mathbf{A}^{(n,l)}\right) \operatorname{vec}\left(\mathbf{X}^{(n)}\right) \\
= \sum_{l=1}^{L} \operatorname{vec}\Big(\Big[\!\!\Big[\mathbf{V}^{(1,l)}, \ldots, \mathbf{V}^{(n-1,l)}, &\mathbf{A}^{(n,l)} \mathbf{X}^{(n)}, \\
&\mathbf{V}^{(n+1,l)}, \ldots, \mathbf{V}^{(N,l)}\Big]\!\!\Big]\Big).
\end{aligned} \tag{4.17}$$

We obtain (4.17) by using (4.6). Next, we compute $\mathbf{y} = \mathbf{J}^{(m)^{\mathrm{H}}} \mathbf{t}$:

$$\begin{aligned}
\mathbf{y} = \mathbf{J}^{(m)^{\mathrm{H}}} \mathbf{t} &= \sum_{l=1}^{L} \left(\mathbf{V}^{\{m,l\}^{\mathrm{H}}} \otimes \mathbf{A}^{(m,l)^{\mathrm{H}}}\right) \mathbf{P}^{(m)} \mathbf{t} \\
&= \sum_{l=1}^{L} \operatorname{vec}\left(\mathbf{A}^{(m,l)^{\mathrm{H}}} \mathbf{T}_{(m)} \mathbf{V}^{\{m,l\}}\right).
\end{aligned} \tag{4.18}$$

We use (4.5) to obtain the last equation, which can be computed efficiently using Tensorlab's `mtkrprod` implementation [215].

Table 4.1: By fully exploiting the Kronecker structure, we obtain a significant improvement in the computational complexity.

		Complexity	
	Calls/iteration	Our algorithm	LS-CPD
Factor matrices $\mathbf{V}^{(n,l)}$	1	$\mathcal{O}(NRIJL)$	/
Objective function	$1 + \text{it}_{\text{TR}}$	$\mathcal{O}(RML)$	$\mathcal{O}(RMI^N)$
Jacobian	1	$\mathcal{O}(NRMLI)$	$\mathcal{O}\left(NRMI^N\right)$
Gradient	1	$\mathcal{O}(NRML)$	$\mathcal{O}(NRMI)$
Gramian-vector	it_{CG}	$\mathcal{O}(NRML)$	$\mathcal{O}(NRMI)$

4.3.5 Block-Jacobi preconditioner

We use a block-Jacobi preconditioner to reduce the number of CG iterations and improve overall convergence. In that case, we have to compute the inverse of ${\mathbf{J}^{(n)}}^{\text{H}}\mathbf{J}^{(n)} \in \mathbb{K}^{RI_n \times RI_n}$, for $1 \leq n \leq N$, in each iteration. The (n,n)th sub-Gramian is given by:

$$ {\mathbf{J}^{(n)}}^{\text{H}}\mathbf{J}^{(n)} = \sum_{l=1}^{L} \left(\mathbf{W}^{\{n,l\}} \otimes {\mathbf{A}^{(n,l)}}^{\text{H}} \mathbf{A}^{(n,l)} \right) \tag{4.19} $$

with $\mathbf{W}^{\{n,l\}} = {\mathbf{V}^{\{n,l\}}}^{\text{H}}\mathbf{V}^{\{n,l\}} \in \mathbb{K}^{R\times R}$ which can be computed as $\mathbf{W}^{\{n,l\}} = *_{q=1,q\neq n}^{N} {\mathbf{V}^{(n,l)}}^{\text{H}}\mathbf{V}^{(n,l)}$ using (4.3).

For $L = 1$, computing the inverse of (4.19) can be done efficiently by omitting the explicit construction of the Jacobians:

$$ \left({\mathbf{J}^{(n)}}^{\text{H}}\mathbf{J}^{(n)}\right)^{\dagger} = \left(\mathbf{W}^{\{n,l\}}\right)^{\dagger} \otimes \left({\mathbf{A}^{(n,l)}}^{\text{H}}\mathbf{A}^{(n,l)}\right)^{\dagger}, $$

in which $\left({\mathbf{A}^{(n,l)}}^{\text{H}}\mathbf{A}^{(n,l)}\right)^{\dagger}$ can be computed beforehand and $\left(\mathbf{W}^{\{n,l\}}\right)^{\dagger}$ requires the inverse of small $(R \times R)$ matrices.

4.3.6 Computational complexity

By exploiting the Kronecker structure in (4.8)-(4.9), we obtain a significant improvement in the computational complexity, enabling an efficient algorithm for large-scale problems, as can be seen in Table 4.1. In order to illustrate this, we compare the per-iteration computational complexity of our algorithm with the LS-CPD algorithm in [26], which ignores the structure of $\mathbf{A}$. For simplicity, we assume that $I_n = I$ and $J_n = J$ for $1 \leq n \leq N$. The number of trust-region (TR) and CG iterations are denoted by it_{TR} and it_{CG}, respectively.

Table 4.2: The block-Jacobi preconditioner (PC) effectively reduces the number of conjugate gradient (CG) iterations in various scenarios. *We reported the average (and standard deviation of the) number of CG iterations across fifty experiments.*

Scenario	No PC	block-Jacobi PC
Square	35 (6)	11 (2)
Underdetermined	38 (7)	13 (2)
Highly underdetermined	60 (0)	35 (6)

4.4 Experiments

4.4.1 The block-Jacobi preconditioner is effective

By using the block-Jacobi preconditioner we can effectively reduce the number of CG iterations in different scenarios. Consider problem (4.8)-(4.9) with $N = 3$, $R = 2$, $L = 3$ $I_1 = I_2 = I_3 = I = 10$, and $K = 1000$. Taking $J_1 = J_2 = J_3 = J$, we consider three scenarios: 1) the square case with $J = 10$ and $M = 1000 = K$, 2) the underdetermined case with $J = 9$ and $M = 729 < K$, and 3) the highly underdetermined case with $J = 5$ and $M = 125 \ll K$. We simulate a typical iteration of the algorithm by computing the Gramian $\mathbf{H}$ and the gradient $\mathbf{g}$ for random factor matrices and then solving $\mathbf{Hp} = -\overline{\mathbf{g}}$ using preconditioned CG until convergence (up to a tolerance of 10^{-6}). We report the average and standard deviation of the number of CG iterations across fifty experiments in Table 4.2 using no PC and the block-Jacobi PC from subsection 4.3.5.

4.4.2 Graph clustering as a Kronecker-structured LS-CPD

Partitioning a graph into meaningful clusters, is crucial to analyze large networks. We show that the similarity-based clustering method in [29] can be reformulated as the computation of a Kronecker-structured LS-CPD. The similarity measure is defined as a weighted infinite sum of the number of common target nodes using neighborhood patterns of any length. It has been shown in [29] that the similarity can then be computed by finding a solution to the following equation:

$$\begin{aligned}\left[\mathbf{I} \otimes \mathbf{I} - \beta^2 \left(\mathbf{G} \otimes \mathbf{G} + \mathbf{G}^{\mathrm{T}} \otimes \mathbf{G}^{\mathrm{T}}\right)\right] \operatorname{vec}\left(\mathbf{S}\right) \\ = \operatorname{vec}\left(\mathbf{G}\mathbf{G}^{\mathrm{T}} + \mathbf{G}^{\mathrm{T}}\mathbf{G}\right)\end{aligned} \tag{4.20}$$

with $\mathbf{G}$ the weighted adjacency matrix of the graph, $\mathbf{S}$ the unknown similarity measure, and a parameter β. In order to reduce the computational cost, it has been proposed in [29] to find a low-rank approximation $\hat{\mathbf{S}}$ of $\mathbf{S}$ instead, reducing (4.20) to (4.8)-(4.9) with $L = 3$ and $N = 2$. In contrast to the method in [29], our approach can easily be extended to $N > 2$, allowing one

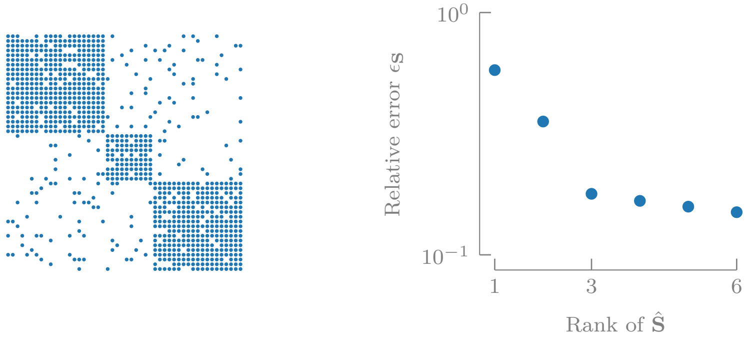


Figure 4.1: A low-rank model $\hat{\mathbf{S}}$ of the similarity measure provides a *good* approximation, allowing one to extract meaningful clusters using this approach [29].

to approximate $\mathbf{S}$ with a low-rank *tensor* model.

We illustrate our method for an Erdős–Rényi random graph with fifty nodes and a simple block structure[1] in the way explained in [29] and visualized in Figure 4.1 (left). In this example, we simulate a community in which nodes primarily interact with other nodes of the same cluster, which occurs, e.g., in (online) social networks. By choosing $R \geq 3$ for this example, we can obtain a meaningful clustering of the nodes because the low-rank model provides a *good* approximation of the underlying similarity measure, as can be seen in Figure 4.1 using relative error $\epsilon_{\mathbf{S}} = ||\mathbf{S} - \hat{\mathbf{S}}||_{\mathrm{F}}/||\mathbf{S}||_{\mathrm{F}}$

4.5 Conclusion

In this chapter, we assumed that the measurement matrix in the LS-CPD paper can be approximated by a sum of a (small) number of Kronecker products. By fully exploiting the structure, we were able to reformulate the LS-CPD problem as a sum of CPDs with linear constraints. This insight allowed us to derive efficient expressions for the ingredients of well-known qN and NLS algorithms, as demonstrated by the complexity analysis, enabling us to tackle large-scale problems. Additionally, we have numerically tested the effectiveness of the block-Jacobi preconditioner and we have demonstrated our approach for graph clustering. In future work, one can derive a more efficient preconditioner for $L > 1$ in order to fully omit the explicit construction of the Jacobians in the algorithm.

---

[1] We us an identity matrix as roll graph and $p_{\mathrm{in}} = 0.9$ and $p_{\mathrm{out}} = 0.1$ [29]. The error results in Figure 4.1 are the median across fifty random experiments.

Part III

Applications

A tensor-based method for large-scale blind source separation using segmentation

5

ABSTRACT | Many real-life signals are compressible, meaning that they depend on much fewer parameters than their sample size. In this chapter we use low-rank matrix or tensor representations for signal compression. We propose a new deterministic method for blind source separation that exploits the low-rank structure, enabling a unique separation of the source signals and providing a way to cope with large-scale data. We explain that our method reformulates the blind source separation problem as the computation of a tensor decomposition, after reshaping the observed data matrix into a tensor. This deterministic tensorization technique is called segmentation and is closely related to Hankel-based tensorization. We apply the same strategy to the mixing coefficients of the blind source separation problem, as in many large-scale applications the mixture is also compressible because of many closely located sensors. Moreover, we combine both strategies, resulting in a general technique that allows us to exploit the underlying compactness of the sources and the mixture simultaneously. We illustrate the techniques for fetal electrocardiogram extraction and direction-of-arrival estimation in large-scale antenna arrays.

This chapter is a slightly adapted version of M. Boussé, O. Debals, and L. De Lathauwer, "A tensor-based method for large-scale blind source separation using segmentation", *IEEE Transactions on Signal Processing*, vol. 65, no. 2, pp. 346–358, Jan. 2017. The figures have been updated for consistency.

5.1 Introduction

In blind source separation (BSS) one tries to reconstruct a set of unobserved sources based only on a set of observed signals. In this chapter, the latter are unknown linear instantaneous mixtures of the sources. Applications can be found in telecommunications, signal processing and biomedical sciences [40], [45], [114], [119]. In general, there is no unique solution to the BSS problem, hence, one imposes additional assumptions.

A well-known BSS method, called independent component analysis (ICA), assumes statistically independent sources [45]. Several ICA methods use higher-order statistics (HOS) in order to tensorize the BSS problem and then apply a tensor decomposition to uniquely identify the sources. Recently, a class of deterministic methods has been proposed that do not use (higher-order) statistics but assume that the sources can be modeled as, e.g., exponential polynomials or rational functions [51], [65]. Specific tensorization techniques can be used, such as Hankel-based or Löwner-based tensorization [62]. The source signals can then be uniquely recovered by block component analysis (BCA). BCA is a framework based on block term decompositions which was introduced in [48], [49], [52]. These methods, as well as the method we propose here, go further than dictionary-based methods. In the latter, one defines *a priori* a fixed signal dictionary in which one assumes the sources can be described sparsely and then one exploits this sparse representability to identify the sources [133], [222]. Here, we do not need an initial dictionary.

In this chapter, we introduce a new method for BSS that exploits the fact that many real-life signals are compressible, i.e., the fact that they can be described in terms of much fewer parameters than the actual number of samples [31], [34]. One way of representing signals in a (possibly very) compact way is a (higher-order) low-rank approximation of a tensorized version of the signal [96]. This can be interpreted as approximating the original signals by sums of Kronecker products of smaller vectors. This strategy is similar to tensor-based scientific computing in high dimensions [96], [152], [214], which has allowed one to solve problems in a number of unknowns that exceeds the number of atoms in the universe. It is used in a novel way for BSS in this chapter and is a key idea to handle large-scale BSS problems, i.e., problems with many sensors and/or samples. In particular, we use a deterministic tensorization technique, called segmentation, that reshapes each observed signal into a matrix (tensor) and stacks them into a (higher-order) tensor. The latter can be interpreted as a compact version of the Hankel-based tensorization mentioned above. We show that the BSS problem boils down to the computation of a decomposition of the tensor obtained by segmentation if the sources exhibit the hypothesized low-rank structure. This yields a unique solution to the BSS problem and provides a way to cope with large-scale problems where conventional methods fall short. Also, it is illustrated that our method allows the separation of underdetermined mixtures, i.e., the separation of more sources than observed signals.

We can apply the same strategy to the mixing coefficients of the BSS problem (instead of the source signals) following a similar argument. Indeed, in the context of big data, we see a large increase in the number of sensors and/or sensor density in fields such as biomedical sciences and sensor array processing [13], [130]. The mixing coefficients are in that case often smoothly varying because of the many closely located sensors, allowing a (higher-order) low-rank approximation of a tensorized version of the mixing vectors. Conventional methods such as ICA fall short in a large-scale setting because of the exponential dependence on the order of the statistics. Exploiting low-rank structure on the mixing level was briefly discussed in [18]. In this chapter, we go further: we apply the strategy on the sources, as described above, but also apply it on both the sources and the mixture simultaneously. The latter is a natural extension that results into a more general method that exploits the hypothesized low-rank structure of the simultaneously tensorized source and mixing level, enabling a unique solution for large-scale BSS.

We illustrate the proposed methods with two applications. First, we have the separation of the fetal and maternal electrocardiogram (ECG) from multilead cutaneous potential recordings. Our method allows a clear separation of the two sources. Second, we have direction-of-arrival (DOA) estimation for large uniform linear arrays in both a line-of-sight and multipath setting. Our methods provides accurate estimates, even for close DOAs. In very large-scale applications, the arrays, however, are typically non-uniform but this is outside the scope of this chapter; here, we focus on the main principles.

In the remainder of this section we introduce the notation and basic definitions. In section 5.2 and section 5.3, we introduce a new BSS method that exploits the hypothesized compressibility of the sources and mixing vectors, respectively. We combine both strategies in section 5.4. Simulations and applications are presented in section 5.5. Finally, we conclude in section 5.6.

5.1.1 Notation and definitions

Tensors, denoted by calligraphic letters (e.g., $\mathcal{A}$), are higher-order generalizations of vectors and matrices, denoted by bold lowercase (e.g., $\mathbf{a}$) and bold uppercase (e.g., $\mathbf{A}$) letters, respectively. The $(i_1, i_2, \ldots, i_N)$th entry of an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$, with $\mathbb{K}$ meaning $\mathbb{R}$ or $\mathbb{C}$, is denoted by $a_{i_1 i_2 \ldots i_N}$. The nth element in a sequence is indicated by a superscript between parentheses (e.g., $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$). The unit vector with a one in the ith row is denoted as $\mathbf{e}_i$.

A mode-n vector of a tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined by fixing every index except the nth, e.g., $\mathbf{a}_{i_1 \cdots i_{n-1} : i_{n+1} \cdots i_N}$, and is a natural extension of the rows and columns of a matrix. The mode-n unfolding of $\mathcal{A}$ is a matrix $\mathbf{A}_{(n)}$ with the mode-n vectors as its columns (following the ordering convention in [125]). The vectorization of $\mathcal{A}$, denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \cdots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1 + \sum_{k=1}^{N}(i_k - 1)J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$. The outer and Kronecker product are denoted by $\otimes$ and $\otimes$, respectively, and are related through a vectorization: $\operatorname{vec}(\mathbf{a} \otimes \mathbf{b}) = \mathbf{b} \otimes \mathbf{a}$. A frontal slice of $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$, denoted by $\mathbf{X}_k$, is obtained by fixing the last index.

5.1.2 Tensor decompositions

An Nth-order tensor has rank one if it can be written as the outer product of N nonzero vectors. The rank of a tensor is defined as the minimal number of rank-1 terms that generate the tensor as their sum. The multilinear rank of an Nth-order tensor is equal to the tuple of mode-n ranks, which are defined as the ranks of the mode-n unfoldings of the tensor.

Definition 24. A *polyadic decomposition* (PD) writes an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ as a sum of R rank-1 terms:

$$\mathcal{A} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(N)}. \tag{5.1}$$

The columns of the factor matrices $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$ are equal to the factor vectors $\mathbf{u}_r^{(n)}$ for $r = 1, \ldots, R$. The PD is called *canonical* (CPD) when R is equal to the rank of $\mathcal{A}$.

The CPD is a powerful model for several applications within signal processing, biomedical sciences, computer vision, data mining and machine learning [41], [125], [168]. The decomposition is *essentially unique* if it is unique up to trivial permutation of the rank-1 terms and scaling and counterscaling of the factors in the same rank-1 term. In general, no unique solution exists in the matrix case without additional assumptions for $R > 1$. In the higher-order case, we typically expect uniqueness under rather mild conditions. Consider a third-order tensor of rank R and size $I \times J \times K$ with factor matrices $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$. Kruskal's condition states that the CPD is unique if [128]:

$$2R + 2 \leq k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}}. \tag{5.2}$$

The k-rank of a matrix $\mathbf{A}$ equals the largest number $k_{\mathbf{A}}$ such that any $k_{\mathbf{A}}$ columns of $\mathbf{A}$ are linearly independent. Condition (5.2) is deterministic in the sense that uniqueness is guaranteed for a particular choice of factor matrices satisfying the condition. Generic uniqueness conditions consider uniqueness with probability one when the entries of the factor matrices are drawn from absolutely continuous probability density functions. For example, condition (5.2) implies generic uniqueness if $2R + 2 \leq \min(I, R) + \min(J, R) + \min(K, R)$ as the k-rank of a generic matrix equals its smallest dimension. In general, milder conditions than Kruskal's can be obtained. Let us for instance consider the case where at least one of the tensor dimensions is not strictly smaller than R. For example, the CPD is generically unique for $\mathbb{K} = \mathbb{C}$ if [38], [73]:

$$R \leq (I-1)(J-1), \quad 3 \leq I \leq J, \quad \text{and} \quad R \leq K. \tag{5.3}$$

More generally, the CPD is generically unique (with a few known exceptions) if [39]:

$$R \leq \left\lceil \frac{IJK}{I+J+K-2} \right\rceil - 1 \quad \text{and} \quad IJK \leq 15000, \tag{5.4}$$

with $\lceil x \rceil$ the smallest integer not less than x. The bound on the number of entries IJK has only been verified numerically up to 15000 but is assumed to hold for larger number of entries as well. Condition (5.4) is equivalent with (5.3) for $R \leq K$ and $3 \leq I \leq J$.

Note that condition (5.4) involves the ratio between the number of entries in the tensor and the number of parameters in a rank-1 term (compensated for scaling). The condition states that the decomposition is unique with probability one if the number of entries is (strictly) larger than the number of parameters, i.e., if the tensor is (minimally) compressible. Our working assumption to solve the large-scale BSS problem is based on this compressibility, as will be explained further. We expect even milder uniqueness conditions when $N > 3$ [166], [184]. An overview and state-of-the-art deterministic and generic uniqueness conditions for higher-order tensors are given in [39], [70]–[75], [135], [184] and references therein. For a short introduction to CPD uniqueness we refer to [168, Section IV].

Definition 25. A *block term decomposition* (BTD) of a third-order tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$ *in multilinear rank-*$(L_r, L_r, 1)$ *terms* for $r = 1, \ldots, R$ is a decomposition of the form:

$$\mathcal{X} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \otimes \mathbf{c}_r, \tag{5.5}$$

in which $\mathbf{A}_r \in \mathbb{K}^{I \times L_r}$ and $\mathbf{B}_r \in \mathbb{K}^{J \times L_r}$ have full column rank L_r and $\mathbf{c}_r$ is nonzero.

These block terms are more general than the simple rank-1 terms of a third-order PD. Hence, they allow the modeling of more complex phenomena, see e.g., [52], [53]. Other types of BTDs and their associated uniqueness results can be found in [48], [49], [51].

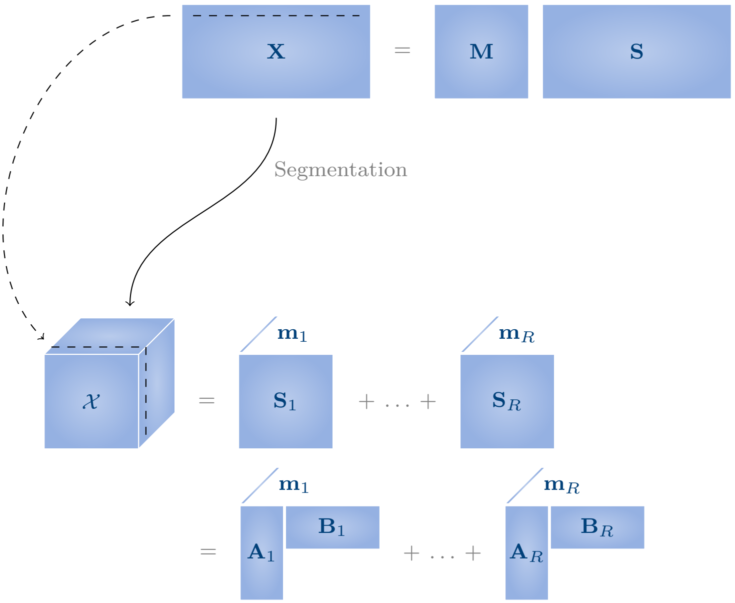


Figure 5.1: Blind Source Separation (BSS) can be reduced to the computation of a block term decomposition (BTD) by means of segmentation if the sources allow a low-rank representation, enabling a unique solution. *Here, each row of the observed data matrix* $\mathbf{X}$ *is reshaped into a matrix and then stacked into a tensor* $\mathcal{X}$*. The reshaped sources appear in the first and second mode, and the mixing vectors appear in the third mode. If the reshaped sources allow a low-rank representation, the BSS problem boils down to a BTD in multilinear rank-*$(L_r, L_r, 1)$ *terms, allowing one to obtain a unique separation of the sources and identification of the mixing vectors.*

5.2 Large-scale blind source separation via low-rank sources

In this section, we derive a new BSS method that exploits the hypothesized compressibility of the sources. We show that this is possible by applying a particular deterministic tensorization technique to the observed data matrix called segmentation. Decomposition of the resulting tensor allows us to uniquely retrieve the mixing vectors and the sources. In subsection 5.2.1, subsection 5.2.2, and subsection 5.2.3, we define BSS, motivate the working hypothesis, and derive our method, respectively.

5.2.1 Blind source separation

We use a linear and instantaneous data model for BSS [45]:

$$\mathbf{X} = \mathbf{MS} + \mathbf{N}, \tag{5.6}$$

with $\mathbf{X} \in \mathbb{K}^{M \times K}$ and $\mathbf{S} \in \mathbb{K}^{R \times K}$ containing K samples of each of the M observed and R source signals, respectively; $\mathbf{M} \in \mathbb{K}^{M \times R}$ is the mixing matrix and $\mathbf{N} \in \mathbb{K}^{M \times K}$ is the additive noise. The goal of BSS is to retrieve the unknown mixing vectors in $\mathbf{M}$ and/or the unknown sources in $\mathbf{S}$, given only the observed data $\mathbf{X}$. In the derivation of our method we ignore the noise $\mathbf{N}$ for notational simplicity, its influence will be further investigated in section 5.5 by means of simulations.

The proposed method reshapes each observed signal, i.e., each row of $\mathbf{X}$, into a matrix and stacks them into a third-order tensor. This is illustrated in Figure 5.1. If the matricized sources admit a low-rank representation, the BSS problem can be solved uniquely by decomposing the tensorized observed data. In general, we reshape each row into an Nth-order tensor and stack them into an $(N+1)$th-order tensor. As such, the parsimonious low-rank models enable very large signal compressions, allowing one to tackle large-scale problems. In general, no unique solution to (5.6) exists without additional assumptions. By assuming that the source signals are low-rank signals, which can be written as sums of Kronecker products of smaller vectors, the problem can be reformulated as a tensor decomposition. As a decomposition of a higher-order tensor is unique under mild conditions as discussed in subsection 5.1.2, the working assumption enables a unique solution of (5.6) under the same conditions.

5.2.2 Low-rank sources

Many real-life signals are compressible, e.g., many common types of signals can be expressed in a basis such that the coefficients decay according to a power law [81]. In a large-scale setting, the amount of information contained in the signal can often be represented by a number of parameters that is much smaller than the total number of entries because there is some structure in the data [190]. Such compressible signals can often be represented in a very compact way by a low-rank approximation of a tensor representation [96], [123]; we call them *low-rank signals*. It is this notion that is the key to our approach: it enables a unique separation of the sources and identification of the mixing vectors. Moreover, it provides a way to cope with large-scale BSS problems because of the large reduction in the number of parameters. We show that our working hypothesis holds exactly for exponential polynomials.

Consider $f(t) = az^t$ evaluated in $t = 0, 1, \ldots, 5$. The resulting vector is reshaped into a (3×2) matrix $\mathbf{S}$ of rank 1:

$$\mathbf{S} = a \begin{pmatrix} 1 & z^3 \\ z & z^4 \\ z^2 & z^5 \end{pmatrix} = a \begin{pmatrix} 1 \\ z \\ z^2 \end{pmatrix} \begin{pmatrix} 1 & z^3 \end{pmatrix}. \tag{5.7}$$

The (3×4) Hankelized version $\mathbf{H}$ of the same vector is [51]:

$$\mathbf{H} = a \begin{pmatrix} 1 & z & z^2 & z^3 \\ z & z^2 & z^3 & z^4 \\ z^2 & z^3 & z^4 & z^5 \end{pmatrix} = \begin{pmatrix} 1 \\ z \\ z^2 \end{pmatrix} \begin{pmatrix} 1 & z & z^2 & z^3 \end{pmatrix}. \tag{5.8}$$

It is well-known that if the original signal is exponential, then $\mathbf{H}$ has rank one, as illustrated. One can see that the columns of $\mathbf{S}$ are a subset of the columns of $\mathbf{H}$. Hence, if $\mathbf{H}$ has rank one, then clearly $\mathbf{S}$ also has rank one. Consider now a vector $\mathbf{f} \in \mathbb{K}^K$ defined by the underlying function $f(t)$ as $f_k = f(t_k)$, $1 \leq k \leq K$, using equidistant samples. We reshape $\mathbf{f}$ into a $(I \times J)$ matrix $\mathbf{S}$ such that $\text{vec}(\mathbf{S}) = \mathbf{f}$ with $K = IJ$. Consider also a Hankelized version $\mathbf{H} \in \mathbb{K}^{I \times J_h}$ such that $h_{ij_h} = f_{i+j_h-1}$ with $K = I + J_h - 1$. Hence, we have that $\mathbf{S} = \mathbf{HQ}$ with $\mathbf{Q} \in \mathbb{K}^{J_h \times J}$ the selection matrix defined by $\mathbf{q}_j = \mathbf{e}_{(j-1)I+1}$ for $j = 1, \ldots, J$. One can verify that the matrix $\mathbf{Q}$ selects all distinct columns of $\mathbf{H}$, by comparing, e.g., the matrices in (5.7) and (5.8). It is clear that if $\mathbf{H}$ has low rank then $\mathbf{S}$ has low rank as well, while $\mathbf{S}$ offers a more compact representation than $\mathbf{H}$. It is known that $\mathbf{H}$ has low rank if the underlying functions are sums of a limited number of exponential and trigonometric terms. This fact extends to the larger class of exponential polynomials [51]. The latter allows one to model a wide range of signals in many applications, e.g., the autonomous behavior of linear systems can be described by (complex) exponential and, if we admit coinciding poles, exponential polynomials. In Table 5.1 we show the coinciding (exact) rank values of $\mathbf{H}$ and $\mathbf{S}$ for several common (exponential) polynomials; by combining such functions one can model a wide variety of signals. For example, a sine is a linear combination of two (complex conjugated) exponentials and, hence, admits a rank-2 model. Note that, while exponential polynomials can be represented by low-rank matrices, the latter allow the representation of a much larger family of signals than only exponential signals. Moreover, Hankel matrices are often ill-conditioned [195], so that the numerical rank can be significantly smaller than the theoretical one.

So far we have discussed signals that admit an exact low-rank representation. However, our approach also works well for more general compressible signals. A reshaped version of the latter often admits an approximate low-rank model as illustrated in Figure 5.2. Assume we approximate $\mathbf{S}$ by a rank-R matrix $\tilde{\mathbf{S}} = \sum_{r=1}^{R} \mathbf{a}_r \otimes \mathbf{b}_r$, then the approximation error on the original function $\mathbf{f} = \text{vec}(\mathbf{S})$ is:

$$||\mathbf{f} - \text{vec}(\tilde{\mathbf{S}})||_{\text{F}}^2 = ||\mathbf{f} - \sum_{r=1}^{R} \mathbf{b}_r \otimes \mathbf{a}_r||_{\text{F}}^2. \tag{5.9}$$

Recall from subsection 5.1.1 that a Kronecker product equals a vectorized outer product. We can make the approximation error (5.9) as small as desired

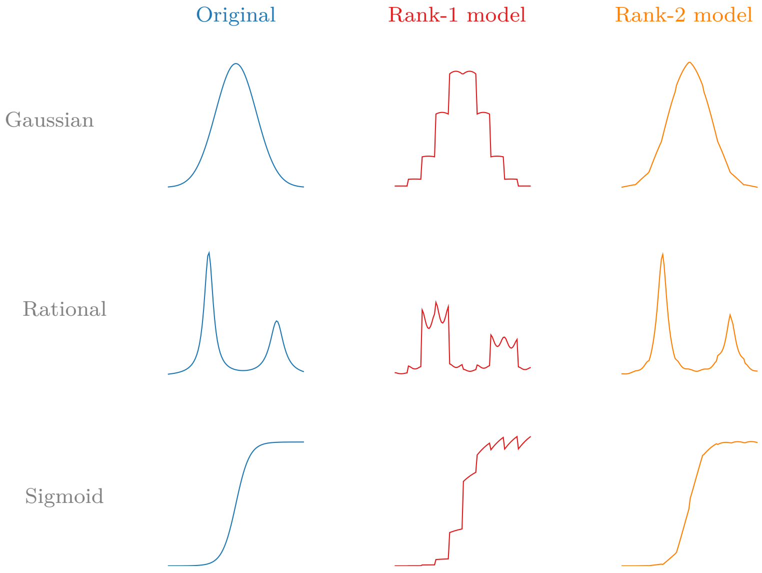


Figure 5.2: A low-rank representation of a reshaped *smooth* function often provides a good approximation. *This is illustrated for a Gaussian, a rational function, and a sigmoid sampled uniformly* 100 *times in* $[0, 1]$. *The original functions are reshaped into a* (10×10) *matrix and then approximated by a low-rank matrix by truncating the singular value decomposition. The reconstructed functions are obtained by vectorizing this low-rank matrix. One can clearly see that the functions can be better approximated by a rank-*2 *rather than a rank-*1 *approximation.*

Table 5.1: If the Hankelized version of an (exponential) polynomial has low rank, then the segmentized version has low rank as well, while the latter provides a more compact version than the former. *We show the rank* $r(\mathbf{H})$ *of the Hankelized version of several (exponential) polynomials* $f(t)$. *If* $\mathbf{H}$ *has low rank then the* $(I \times J)$ *reshaped version* $\mathbf{S}$ *has low rank as well (if* $R < min(I, J)$*). The latter, however, provides a much more compact representation for* $f(t)$ *than the former. (*$p_r(t)$ *is a polynomial of degree* Q_r*.)*

$f(t)$	$r(\mathbf{H})$	$f(t)$	$r(\mathbf{H})$
az^t	1	$\sum_{r=1}^{R} a_r z_r^t$	R
$a\sin(bt)$ $a\cos(bt)$	2	$\sum_{r=1}^{R} a_r \sin(b_r t)$	$2R$
$az^t \sin(bt)$	2	$\sum_{r=1}^{R} a_r z_r^t \sin(b_r t)$	$2R$
$p(t) = \sum_{q=0}^{Q} a_q t^q$	$Q+1$	$\sum_{r=1}^{R} p_r(t)$	$\sum_{r=1}^{R} Q_r + R$
$p(t)z^t$	$Q+1$	$\sum_{r=1}^{R} p_r(t) z_r^t$	$\sum_{r=1}^{R} Q_r + R$
$p(t)\sin(at)$	$2Q+2$	$\sum_{r=1}^{R} p_r(t)\sin(a_r t)$	$\sum_{r=1}^{R} Q_r + 2R$
$p(t)z^t \sin(at)$	$2Q+2$	$\sum_{r=1}^{R} p_r(t) z_r^t \sin(a_r t)$	$\sum_{r=1}^{R} Q_r + 2R$

by increasing R. Since (5.9) is just a vectorized version of $\|\mathbf{S} - \tilde{\mathbf{S}}\|_{\mathrm{F}}^2$, Eckart–Young's theorem provides an upper bound on the approximation error [80]. Namely, the least-squares error on the representation of the signal $\mathbf{f}$ is the sum of the squares of the discarded singular values of $\mathbf{S}$. The singular value spectrum of $\mathbf{S}$ is often fast decaying, and hence the signal $\mathbf{f}$ often admits a good representation of the form (5.9) for low R. It is outside the scope of this chapter to investigate in general under which conditions on the signal $\mathbf{f}$ the error in (5.9) is small. However, we do provide explicit bounds by focusing on signals that admit a good polynomial approximation. We emphasize that these are only bounds, as 1) polynomials are only a special case of exponential polynomials and 2) the latter are only a special case of functions that yield a low-rank matrix $\mathbf{S}$. As such, assume that we approximate the underlying function $f(t)$ of $\mathbf{f}$ with a Taylor polynomial $p(t)$ of degree $R-1$ around $t = t_*$. Assuming $f(t)$ and its derivatives up to order R are continuous, which is satisfied for smooth signals, Taylor's theorem provides the following element-wise upper bound on the error in (5.9):

$$|f(t) - p(t)| \leq \frac{f_{\max}}{R!} |t - t_*|^R \tag{5.10}$$

with $f_{\max} = \max_u f^{(R)}(u)$, $u \in (t_*, t)$ and $f^{(R)}$ the Rth derivative of f. The corresponding matrix $\tilde{\mathbf{S}}$ of $p(t)$ has rank R, see Table 5.1; hence, (5.10) is a bound on the error of the rank-R approximation of $\mathbf{S}$. Signals with rapidly converging Taylor series admit an approximate low-rank model, hence, only a small R is needed for a good approximation. A general polynomial approximation $p(t)$ in K uniformly sampled points in the interval $[a, b]$ gives the following upper bound on (5.9):

$$||\mathbf{f} - \mathrm{vec}(\tilde{\mathbf{S}})||_{\mathrm{F}}^2 \leq \left(\frac{h^R}{4R} f_{\max}\right)^2$$

with $h = (b-a)/R$, $\bar{f} = \max_u f^{(R)}(u)$, $u \in [a, b]$, and $f^{(R)}$ the Rth derivative of f. Similar results can be derived for other types of approximations, e.g., a polynomial approximation in Chebyshev points. In section 5.5, we illustrate our strategy for real-life signals as well, showing that our working hypothesis is valid for a variety of signals and applications.

In this chapter we also reshape signals into higher-order tensors, going further than the Hankel strategy from [51] and enabling an even more compact representation. In tensor-based scientific computing one often reshapes a function up to a $(2 \times 2 \times \cdots \times 2)$ tensor of very high order to achieve maximal compression for a fixed rank R [96], [123]. Here, we allow much more freedom in the choice of the reshaping parameters, which enables a trade-off between the approximation error in (5.9) and the compression rate, see subsection 5.5.5.

Let us now describe the strategy more formally. Suppose one reshapes the rth source $\mathbf{s}_r$ in (5.6) into a $(I \times J)$ matrix $\mathbf{S}_r$ such that $\mathrm{vec}(\mathbf{S}_r) = \mathbf{s}_r$ with $K = IJ$. Note that this is the same as stacking different decimated versions of the signal in the rows of a matrix. If the rth reshaped (or matricized) source $\mathbf{S}_r$ admits a rank-1 representation, which is our working hypothesis, we have that $\mathbf{S}_r = \mathbf{a}_r \otimes \mathbf{b}_r$ with $\mathbf{a}_r \in \mathbb{K}^I$ and $\mathbf{b}_r \in \mathbb{K}^J$, as, e.g., in (5.7). In general, however, this model is too restrictive. The reshaped sources may admit, or better be approximated by a low-rank representation, as is, e.g., the case for a sine and the functions in Figure 5.2, respectively. Hence, we have that $\mathbf{S}_r = \sum_{l_r=1}^{L_r} \mathbf{a}_{l_r r} \otimes \mathbf{b}_{l_r r}$. Note that this means that we assume that the sources can be written as a sum of Kronecker products: $\mathbf{s}_r = \mathrm{vec}\left(\mathbf{S}_r\right) = \sum_{l_r=1}^{L_r} \mathbf{b}_{l_r r} \otimes \mathbf{a}_{l_r r}$. This strategy enables a compact representation of the sources, see Table 5.2. Indeed, the number of parameters is one order of magnitude lower than the finite sample length K if $I \approx J$.

More generally, we can reshape the sources into a higher-order tensor, enabling a more compact representation. Suppose we reshape the rth source $\mathbf{s}_r$ into an Nth-order tensor $\mathcal{S}_r \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ such that $\mathrm{vec}(\mathcal{S}_r) = \mathbf{s}_r$ with $K = \prod_{n=1}^{N} I_n$. If the rth reshaped (or tensorized) source $\mathcal{S}_r$ admits a (higher-

Table 5.2: Segmentation enables possibly large compression ratios, indicating the applicability of this strategy for large-scale BSS. *By reshaping* $\mathbf{s}_r$ *in* (5.6) *into* $\mathcal{S}_r \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ *and then using a rank-*L_r *representation leads to a considerable compression. If* $N = 2$*, we use* I *and* J*. The number of parameters decreases logarithmically in* N *and increases proportionally with* L_r*.*

	K	for general I_n	$I_n \approx I$, for all n
$N = 2$	IJ	$L_r(I + J - 1)$	$\mathcal{O}(L_r I)$
$N > 2$	$\prod_{n=1}^{N} I_n$	$L_r(\sum_{n=1}^{N} I_n - N + 1)$	$\mathcal{O}(L_r N I)$

order) low-rank representation, we have that:

$$\mathcal{S}_r = \sum_{l_r=1}^{L_r} \mathbf{u}_{l_r r}^{(1)} \otimes \mathbf{u}_{l_r r}^{(2)} \otimes \cdots \otimes \mathbf{u}_{l_r r}^{(N)}, \tag{5.11}$$

in which $\mathbf{u}_{l_r r}^{(n)} \in \mathbb{K}^{I_n}$ for $n = 1, \ldots, N$, where the number of rank-1 terms L_r can differ between sources. Note that this is a PD as in (5.1). This means that the sources can be modeled, or approximated, by sums of $(N-1)$ Kronecker products [96]:

$$\mathbf{s}_r = \operatorname{vec}(\mathcal{S}_r) = \sum_{l_r=1}^{L_r} \mathbf{u}_{l_r r}^{(N)} \otimes \mathbf{u}_{l_r r}^{(N-1)} \otimes \cdots \otimes \mathbf{u}_{l_r r}^{(1)}, \tag{5.12}$$

In general, the number of parameters decreases exponentially in the number of Kronecker products N (i.e., the order of the representation) and increases proportionally with the number of rank-1 terms L_r, see Table 5.2. For example, if $I_n = I$ for $n = 1, \ldots, 3$, then $K = I^3$ and only $\mathcal{O}(3L_r I)$ parameters are needed. The possibly large compressions indicate the applicability of this strategy for large-scale BSS problems.

5.2.3 Decomposition

We now demonstrate how the BSS problem in (5.6) can be reformulated as the computation of a tensor decomposition when the sources admit a low-rank representation. Let us start as follows: each row of $\mathbf{X}$ is reshaped into a $(I \times J)$ matrix as described earlier and then stacked into a third-order tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times M}$ such that $\operatorname{vec}(\mathbf{X}_m) = \mathbf{x}_m$. In other words, the mth matricized observed signal is equal to the mth frontal slice of $\mathcal{X}$. Since the tensorization is a linear operation, the M reshaped observed signals are linear combinations of the R reshaped sources $\mathbf{S}_r \in \mathbb{K}^{I \times J}$. As such, we have that:

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{S}_r \otimes \mathbf{m}_r \tag{5.13}$$

with $\mathbf{m}_r$ the rth column of $\mathbf{M}$. We denote this deterministic tensorization technique by *segmentation*; see Figure 5.1 for an illustration. Now assume that the rth reshaped source in (5.13) admits a rank-1 representation, i.e., $\mathbf{S}_r = \mathbf{a}_r \otimes \mathbf{b}_r$ for $r = 1, \ldots, R$, then we have that:

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \otimes \mathbf{b}_r \otimes \mathbf{m}_r. \tag{5.14}$$

Equation (5.14) is a CPD as defined in (5.1). Consequently, the BSS problem boils down to the computation of a CPD of a third-order tensor in R rank-1 terms. Analogously, if the reshaped sources admit a low-rank representation, the BSS problem boils down to a BTD in multilinear rank-$(L_r, L_r, 1)$ terms, as in (5.5) and illustrated in Figure 5.1. References to uniqueness results for both cases have been mentioned in subsection 5.1.2. We insist that the compressibility of the sources has enabled their blind separation.

More generally, we can reshape each observed signal into a $(I_1 \times I_2 \times \cdots \times I_N)$ Nth-order tensor as described earlier and then stack it into a $(N+1)$th-order tensor $\mathcal{X} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N \times M}$. As such, the mth tensorized observed signal is equal to the mth Nth-order "frontal slice" of $\mathcal{X}$:

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{S}_r \otimes \mathbf{m}_r, \tag{5.15}$$

If the reshaped sources $\mathcal{S}_r \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ allow a low-rank representation as in (5.11), we have:

$$\mathcal{X} = \sum_{r=1}^{R} \left(\sum_{l_r=1}^{L_r} \mathbf{u}_{l_r r}^{(1)} \otimes \mathbf{u}_{l_r r}^{(2)} \otimes \cdots \otimes \mathbf{u}_{l_r r}^{(N)} \right) \otimes \mathbf{m}_r, \tag{5.16}$$

which is a decomposition in R (rank-L_r $\otimes$ vector) terms [180]. It is a more general decomposition because it boils down to a CPD of a higher-order tensor as in (5.1) if $L_r = 1$ for all r. Also, it boils down to a BTD in multilinear rank-$(L_r, L_r, 1)$ terms as in (5.5) if $N = 2$, i.e., if $\mathcal{X}$ is a third-order tensor. In that case, the factor matrices $\mathbf{U}_r^{(1)}$ and $\mathbf{U}_r^{(2)}$ of the rth term are not unique, but their products are (up to scaling and permutation). On the other hand, for $N > 2$, the factor matrices $\mathbf{U}_r^{(n)}$ are unique under mild conditions because they form a rank-L_r PD of an Nth-order tensor. We will exploit this in the DOA estimation application in subsection 5.5.8.

The proposed method simultaneously determines both the mixing vectors *and* the sources by 1) simply reshaping the data (using segmentation) and 2) exploiting the fact that many real-life signals admit a (higher-order) low-rank representation. As such, the BSS problem boils down to a tensor decomposition and 3) we can benefit from mild uniqueness properties. Moreover, 4) it is applicable for large-scale BSS problems, i.e., large K, as is clear from the possibly huge compressions as indicated above. However, this is not necessarily a significant advantage compared to existing methods like ICA. The latter has only a linear dependence on K and even benefits from large K accuracy-wise because the K samples are used to estimate statistics. Finally, 5) the method is deterministic, meaning that it does not use (higher-order) statistics, hence, it also works well if the number of samples is small and/or if the sources are not statistically independent. This is a difference with statistical methods such as ICA.

5.3 Large-scale blind source separation via low-rank mixing vectors

In the previous section we exploited the fact that many real-life (source) signals admit a low-rank representation. This is also a natural assumption for the mixing vectors if one considers, e.g., many sensors and/or high sensor density; we call them *low-rank mixing vectors* analogous to low-rank sources. Such problems arise in biomedical sciences, e.g., wireless body area networks (WBANs) using electroencephalography (EEG) [13] and electrocorticography (ECoG) [161] with high spatial resolution, or neural dust with thousands of miniature sensors (neural probes) dispersed throughout the brain [163]. Moreover, one often encounters mixing matrices with Vandermonde structure [181], i.e., each reshaped mixing vector has exactly rank one. An example are uniform linear (ULAs) and rectangular arrays (URAs) with far-field sources that emit narrowband signals [127], [141], [167]. Here, we also see a trend towards large-scale antennas, also known as massive MIMO [86], [130]. If the signals propagate through several distinct paths, e.g., due to reflections or scattering [118], each reshaped mixing vector has low rank. If the sources are located in the near-field, the Vandermonde structure is only approximate which can be accommodated by a low-rank approximation.

Exploitation of the underlying compactness of such low-rank mixing vectors amounts to a comparable method as section 5.2, which has been briefly addressed in [18]. Let us illustrate the analogy with the previous section more clearly: each *column* (cf. above) of $\mathbf{X}$ is reshaped into a $(I \times J)$ matrix with $M = IJ$ and then stacked into a third-order tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$. Next, assume the reshaped mixing vectors admit a rank-1 representation, which is our working hypothesis, i.e., $\mathbf{M}_r = \text{unvec}(\mathbf{m}_r) = \mathbf{a}_r \otimes \mathbf{b}_r$ for $r = 1, \ldots, R$.

Hence, we obtain:

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \otimes \mathbf{b}_r \otimes \mathbf{s}_r. \tag{5.17}$$

Note that this boils down to applying the same strategy as before on the transposed observed data matrix. The generalization to higher-order low-rank representations is straightforward. The same analysis as in subsection 5.2.3 applies, but now we segment the mixing vectors and exploit the fact that they possibly admit a (higher-order) low-rank representation. Moreover, the method has several advantages over ICA: ICA methods based on (full) HOS are infeasible when M is large as the number of entries in Qth-order statistics is $\mathcal{O}(M^Q)$. Also, our method can handle Gaussian random sources in contrast to ICA (if the mixing vectors indeed exhibit some low-rank structure) [45]. Finally, the method imposes only mild conditions (via the uniqueness conditions) on the sources in contrast to existing methods, e.g., linear independence instead of statistical independence as in ICA.

5.4 Large-scale blind source separation using twofold segmentation

In the previous two sections we either reshaped the sources *or* the mixing vectors and then exploited the hypothesized low-rank structure. However, as we have illustrated before, both the mixing vectors *and* the sources may admit such a higher-order low-rank representation. Hence, a natural extension is to use both strategies simultaneously. For instance, one often has sinusoidal sources, which admit a rank-2 representation, in ULAs of which the Vandermonde mixing vectors admit a rank-1 representation. To the best of our knowledge, this is the first time that tensorization is used on both levels of the BSS problem and more generally in matrix factorization.

By exploiting the underlying compactness on both levels, we are again able to reformulate the BSS problem as the computation of a tensor decomposition. Let us start with reshaping each column of $\mathbf{X}$ into a $(I_1 \times I_2)$ matrix with $M = I_1 I_2$ and stacking them in an intermediate third-order tensor $\mathcal{Y} \in \mathbb{K}^{I_1 \times I_2 \times K}$. Note that the (i_1, i_2)th mode-3 vector of $\mathcal{Y}$ equals the $(i_1 + (i_2 - 1)I_1)$th row of $\mathbf{X}$. Each mode-3 vector of $\mathcal{Y}$ (i.e., row of $\mathbf{X}$) is subsequently reshaped into a $(J_1 \times J_2)$ matrix with $K = J_1 J_2$, which overall yields a fourth-order tensor $\mathcal{X} \in \mathbb{K}^{I_1 \times I_2 \times J_1 \times J_2}$. Hence, we have that:

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{M}_r \otimes \mathbf{S}_r. \tag{5.18}$$

We denote this by *twofold segmentation* (cf. section 5.2 and section 5.3). Let us now assume that both the reshaped mixing vectors and sources admit a rank-1 representation. In that case, it is easy to see that (5.18) is a CPD of a fourth-order tensor in R rank-1 terms. More generally, if the segmented mixing vectors and sources allow a low-rank representation, we have:

$$\mathcal{X} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \otimes (\mathbf{C}_r \mathbf{D}_r^{\mathrm{T}}), \tag{5.19}$$

in which $\mathbf{A}_r \in \mathbb{K}^{I_1 \times L_r}$ and $\mathbf{B}_r \in \mathbb{K}^{I_2 \times L_r}$ have full column rank L_r and $\mathbf{C}_r \in \mathbb{K}^{J_1 \times P_r}$ and $\mathbf{D}_r \in \mathbb{K}^{J_2 \times P_r}$ have full column rank P_r. Note that the ranks L_r and P_r can be different for each r and do not necessarily have the same value inside the rth term. This is a new kind of decomposition: $\mathcal{X}$ is decomposed in a sum of R (rank-L_r $\otimes$ rank-P_r) terms.

More generally, we can reshape each row and column of $\mathbf{X}$ into $\mathcal{S}_r \in \mathbb{K}^{J_1 \times J_2 \times \cdots \times J_{N_s}}$ and $\mathcal{M}_r \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_{N_m}}$ such that $\mathrm{vec}(\mathcal{M}_r) = \mathbf{m}_r$ and $\mathrm{vec}(\mathcal{S}_r) = \mathbf{s}_r$, respectively, with $M = \prod_{n_m=1}^{N_m} I_{n_m}$ and $K = \prod_{n_s=1}^{N_s} J_{n_s}$, analogous to the single segmentation case in (5.15). As such, we have that:

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{M}_r \otimes \mathcal{S}_r.$$

Analogous to (5.16), the reshaped mixing vectors and sources can both admit a low-rank representation. Hence, we have that:

$$\mathcal{X} = \sum_{r=1}^{R} \left(\sum_{l_r=1}^{L_r} \otimes_{n_m=1}^{N_m} \mathbf{u}_{l_r r}^{(n_m)} \right) \otimes \left(\sum_{p_r=1}^{P_r} \otimes_{n_s=1}^{N_s} \mathbf{v}_{l_r r}^{(n_s)} \right),$$

in which $\mathbf{u}_{l_r r}^{(n_m)} \in \mathbb{K}^{I_{n_m}}$ and $\mathbf{v}_{p_r r}^{(n_s)} \in \mathbb{K}^{J_{n_s}}$. In comparison with (5.19), the block factors $\mathbf{U}_r^{(n)}$ and/or $\mathbf{V}_r^{(n)}$ are unique under mild conditions if $N_m > 2$ and/or $N_s > 2$. The reason is the same as for the single segmentation case, see subsection 5.2.3.

The proposed method offers 1) a framework to exploit the low-rank structure of both the reshaped mixing vectors and sources; the same analysis as in the previous sections applies. Again, we reformulate the BSS problem as the computation of a tensor decomposition, hence, 2) we can benefit from the mild uniqueness properties. More specifically, it boils down to the computation of a new and more general decomposition. As such, 3) the method is applicable in a big data setting: it can handle both large sample sizes and large numbers of sensors efficiently, see Table 5.2. Furthermore, 4) the method is deterministic, hence, it is not needed *per se* to have a large number of samples. Finally, 5) only mild, and natural, assumptions are imposed on the mixing vectors and the sources. We simply exploit the low-rank structure which is often present in real-life signals as explained above.

5.5 Simulations and applications

In subsection 5.5.1, we give an example of the separation of two low-rank sources and the separation of two low-rank sources that are mixed with low-rank mixing vectors. In subsection 5.5.2, we demonstrate the separation of more sources than observed signals. We investigate the influence of noise and sample size in subsection 5.5.3. In subsection 5.5.4, we show how well one can approximate the reshaped mixing vectors and/or sources for varying rank and signal-to-noise ratio (SNR). In subsection 5.5.5, we analyze the influence of the choice of reshaping dimensions. In subsection 5.5.6, we analyze consistency numerically. Finally, in the last two subsections, we illustrate the proposed methods with fetal electrocardiogram extraction and direction-of-arrival estimation in large-scale uniform linear arrays.

We use the `segmentize` command from Tensorlab to apply segmentation to the observed data matrices [215]. The CPD and BTD in multilinear rank-$(L_r, L_r, 1)$ terms can typically be computed algebraically by means of a generalized eigenvalue decomposition [49], [178], [180]. The algebraic solution is exact in the noiseless case and a good initialization for optimization-based methods in the noisy case. In this chapter, we use least-squares (LS) optimization-based algorithms `cpd` and `ll1` to fit the decomposition to the data until a sufficiently high accuracy is attained. During the computation, it is theoretically possible that degeneracy occurs [126], [172]. For example, the magnitude of some terms grows without bounds but with opposite sign, resulting in a poor solution but a good fit. Degeneracy can be avoided in several ways such as increasing the number of rank-1 terms or imposing orthogonality or non-negative constraints on the factor matrices [41], [126], [134], [185]. The decompositions in (rank-$L_r \otimes$ vector) and (rank-$L_r \otimes$ rank-P_r) terms are computed with two adapted versions of `cpd_nls` called `lvec_nls` and `lp_nls`, respectively, and are available upon request. For very large tensors, one can resort to large-scale algorithms as described in [171], [209], [214].

The mixing vectors and sources can only be determined up to scaling and permutation, i.e., the standard indeterminacies in BSS. Hence, in order to compute the error they are first optimally scaled and permuted with respect to the true ones. The relative error is then defined as the relative difference in Frobenius norm, i.e., we have relative error $\epsilon_{\mathbf{A}} = ||\mathbf{A} - \hat{\mathbf{A}}||_{\mathrm{F}}/||\mathbf{A}||_{\mathrm{F}}$ with $\hat{\mathbf{A}}$ an optimally scaled and permuted estimate of $\mathbf{A}$.

We use additive i.i.d. Gaussian noise unless indicated otherwise. Note that existing optimization-based algorithms for computing a tensor decomposition typically employ a LS cost function, which is optimal (in the maximum likelihood sense) when the data is perturbed by additive i.i.d. Gaussian noise. In order to efficiently handle nonidentically distributed errors in the LS setting, we use a low-rank weight tensor in Appendix A. For other types of noise distributions, one can use cost functions that employ β-divergences [200], which are a type of divergence that interpolate between the LS distance ($\beta = 2$), Kullback–Leibler divergence ($\beta = 1$), and Itakura–Saito divergence ($\beta = 0$) [105]. Errors on small (large) entries are penalized more than in the LS setting for $\beta < 2$ ($\beta > 2$), rendering β-divergences especially useful for data data with entries of different magnitudes. This approach has been used successfully in nonnegative matrix factorization (NMF) of audio spectra [85].

5.5.1 General experiments

First, we illustrate the method proposed in section 5.2. Consider $R = 2$ low-rank sources: $s_1(t) = e^{-t}$ and $s_2(t) = \sin(4\pi t)$ with $K = 4096$ equidistant samples in $[0, 1]$. They are mixed into $M = 3$ observed signals using $\mathbf{M} = [0.5,\ 2;\ 2,\ -3;\ 1,\ 0.5]$. We use a second-order ($N = 2$) rank-1 ($L_1 = 1$) and rank-2 ($L_2 = 2$) approximation for the first and second source, respectively, with $I = J = 64$. Note that the approximation of the first and second source requires only 127 and 254 values, respectively, see Table 5.2. This is the maximal reduction for a second-order approximation. Namely, we have a compression of $1 - L_r\frac{I+J-N+1}{M}$, i.e., 96.90% and 93.80% for the first and second source, respectively. The perfectly recovered sources are shown in Figure 5.3.

Second, we illustrate the method proposed in section 5.4. Consider $R = 2$ low-rank sources: $s_1(t) = e^{-t} + e^{t} - e^{0.5t}$ and $s_2(t) = 2e^{-t}$ with $K = 4096$ equidistant samples in $[0, 1]$. The sources are mixed with two low-rank mixing vectors: $m_1(\xi) = \sin(2\pi\xi)$ and $m_2(\xi) = e^{-2\xi}\sin(6\pi\xi)$ with $M = 4096$ equidistant samples in $[0, 1]$. We use a third-order ($N_s = 3$) rank-3 ($P_1 = 3$) and rank-1 ($P_2 = 1$) approximation for the first and second source, respectively, with $J_1 = J_2 = J_3 = 16$. Furthermore, we use a second-order ($N_m = 2$) rank-2 approximation for both mixing vectors ($L_1 = L_2 = 2$) with a non-optimal choice of the segmentation parameters: $I_1 = 128$ and $I_2 = 32$. Hence, we decompose the $(128 \times 32 \times 16 \times 16 \times 16)$ segmented version of $\mathbf{X}$ into a sum of a (rank-2 $\otimes$ rank-3) and a (rank-2 $\otimes$ rank-1) term. The approximation of the rth mixing vector requires only $L_r(I_1 + I_2 - N_m + 1)$ values, i.e., a compression of $1 - L_r\frac{I_1+I_2-N_m+1}{M} = 92.19\%$, although this is not the maximal compression. Higher compression can be attained by increasing the order. For instance, the approximation of the rth source consists of only $P_r(J_1 + J_2 + J_3 - N_s + 1)$ values, i.e., a compression of $1 - P_r\frac{J_1+J_2+J_3-N_s+1}{M}$. Specifically, we have a compression of 96.63% and 98.88% for the first and second source, respectively. We further investigate the choice of I_{n_m} and J_{n_s} in subsection 5.5.5. The perfectly recovered factors are shown in Figure 5.4.

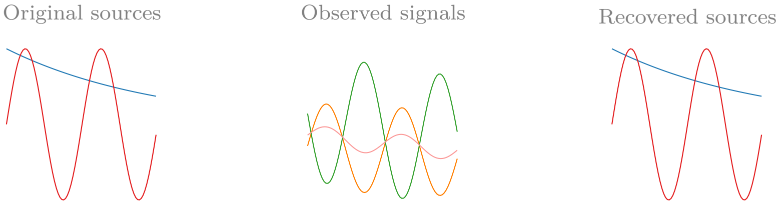


Figure 5.3: By exploiting the intrinstic low-rank structure of the sources, the source signals can be perfectly reconstructed (in the noiseless case).

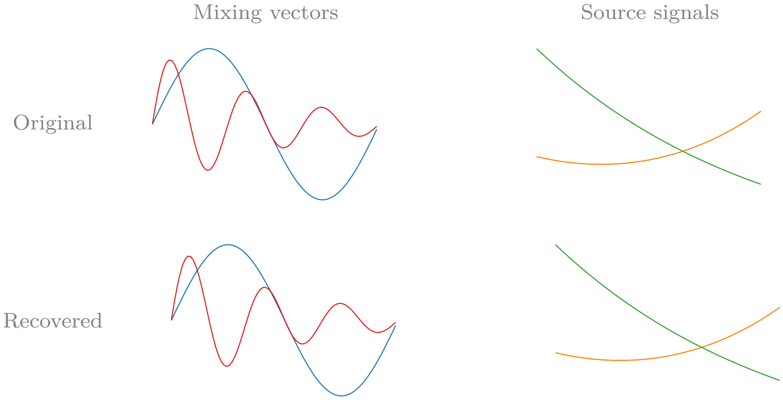


Figure 5.4: By exploiting the low-rank structure of the mixing coefficients *and* the sources, the mixing vectors and the source signals are perfectly reconstructed (in the noiseless case).

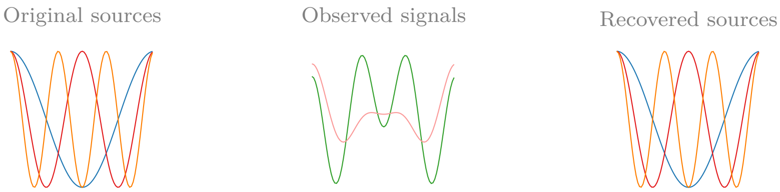


Figure 5.5: By exploiting the intrinstic low-rank structure of the sources, the source signals can even be perfectly recovered from *underdetermined* mixtures (in the noiseless case).

5.5.2 Underdetermined mixture

We illustrate the separation of more sources than observed signals. Consider $R = 3$ complex exponential source signals $s_r(t) = e^{2\pi irt}$ for $r = 1, \ldots, R$ which are mixed into $M = 2$ observed signals using a mixture matrix $\mathbf{M} = [-1,\ 0.5,\ 2;\ 0.5,\ 1,\ 0.5]$. We take $K = 4096$ uniformly discretized samples in $[0, 1]$. We use a second-order ($N = 2$) rank-1 approximation for the sources with $I_1 = I_2 = 64$. The real part of the recovered sources is shown in Figure 5.5: perfect reconstruction is obtained.

5.5.3 Noise and sample length

First, we investigate the influence of the noise and the sample size K for the method of section 5.3. Consider a setup in which we have $M = 4096$ sensors and $R = 2$ i.i.d. zero-mean unit-variance Gaussian random sources of length $K = \{10^1, 10^2, 10^3\}$. We construct the low-rank mixing vectors as the vectorization of a second-order ($N = 2$) rank-2 ($L_1 = 2$) and rank-3 ($L_2 = 3$) tensor using (5.12) with zero-mean unit-variance Gaussian random factor vectors and $I = J = 64$. Hence, we use a second-order rank-2 and rank-3 approximation with $I = J = 64$, respectively. In Figure 5.6, we report the relative error on the mixing vectors $\epsilon_{\mathbf{M}}$ and the sources $\epsilon_{\mathbf{S}}$; note that the results are very accurate in comparison with the SNR. Although the method is deterministic, it is beneficial to increase K under noisy conditions. However, K can be (very) low in comparison to typical values in ICA. (Note that in this particular example, ICA cannot be used since the sources are Gaussian.) $\epsilon_{\mathbf{S}}$ does not improve for increasing K because one also has to estimate longer source signals. Similar results can be obtained for the method of section 5.2 when increasing the number of sensors M under noisy conditions.

Next, consider a similar setup as in the previous experiment but now with the following rank-1 mixing vectors: $m_1(\xi) = e^{0.5\xi}$ and $m_2(\xi) = e^{-2\xi}$ with $\xi \in [0, 1]$. We use a second-order ($N = 2$) rank-1 approximation for both mixing vectors ($L_1 = L_2 = 1$) with $I = J = 64$. The results are shown in Figure 5.7: in comparison with Figure 5.6, there is some loss of accuracy

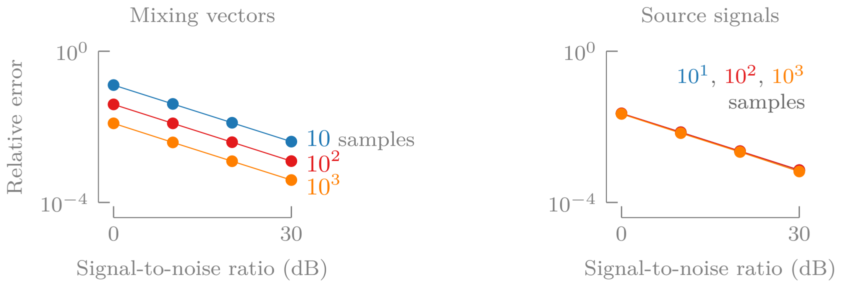


Figure 5.6: By increasing the number of samples, the relative error on the mixing vectors can be improved. This does not hold for the relative error on the sources because we also have to estimate longer source signals. Eventhough, the number of samples can be low in contrast to stochastic methods, the results are very accurate compared to the signal-to-noise ratio. *Here, we plot the median across* 100 *experiments of the relative error on the mixing vectors and the sources as a function of the signal-to-noise ratio for* 10, 10^2, *and* 10^3 *samples. The mixing vectors are well conditioned; compare with Figure* 5.7.

on the mixing vectors and much clearer on the sources. This is due to the condition of the problem: in the previous experiment, the mixing vectors are approximately orthogonal and have about the same size ($\|\mathbf{m}_1\|/\|\mathbf{m}_2\| \approx 0.8$), while now the angle is 37.11° and $\|\mathbf{m}_1\|/\|\mathbf{m}_2\| = 2.65$. Hence, the computation of the decomposition is more difficult and the estimates less accurate.

5.5.4 Low-rank approximation

We investigate the influence of deviations from a second-order rank-1 structure on the relative error as follows. Define each mixing vector as the vectorization of a random matrix with exponentially decaying singular values, i.e., $\mathbf{m_r} = \operatorname{vec}\left(\mathbf{U_r}\operatorname{diag}\left(\boldsymbol{\sigma}\right)\mathbf{V_r}\right)$ with $\boldsymbol{\sigma} = e^{-\alpha\boldsymbol{\xi}}$ and $\boldsymbol{\xi}$ a vector containing $\min\left(I, J\right)$ equidistant samples in $[0, 1]$. $\mathbf{U}_r$ and $\mathbf{V}_r$ are random orthogonal matrices of compatible dimensions. The exponential decay of the singular values is controlled with α which is a measure for the rank-1-ness of the mixing vectors: increasing α leads to more rank-1-like mixing vectors and vice-versa. We take $R = 2$ i.i.d. zero-mean unit-variance Gaussian random sources of length $K = 10$ and use a second-order ($N = 2$) rank-L_r approximation with $I = J = 64$.

Figure 5.8 shows the relative errors $\epsilon_{\mathbf{M}}$ and $\epsilon_{\mathbf{S}}$ as a function of α for an SNR of 15 dB using $L_1 = L_2 = L = 1$. Note that an estimate of the mixing matrix $\hat{\mathbf{M}}$ can be obtained from the decomposition, i.e., from (5.17) for this particular case, in the way explained above. However, one can also estimate it via the noisy observed data matrix and the pseudo-inverse of the

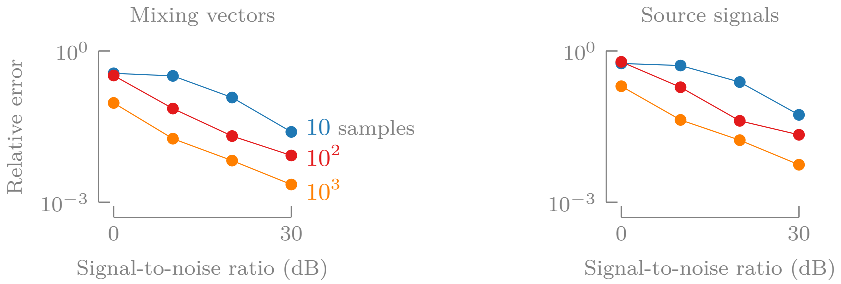


Figure 5.7: By using ill-conditioned mixing vectors, we simulate a *difficult* problem, hence, the computation is more difficult and the estimates are less accurate, as shown above. The effect is more pronounced for the source signals than the mixing vectors. *Here, we plot the median across* 100 *experiments of the relative error on the mixing vectors and the sources as a function of the signal-to-noise ratio for* 10*,* 10^2*, and* 10^3 *samples. The mixing vectors are ill-conditioned; compare with Figure* 5.6*.*

estimated source matrix: $\hat{\mathbf{M}} = \mathbf{X}\hat{\mathbf{S}}^\dagger$. The figure illustrates that $\epsilon_{\mathbf{M}}$ decreases for increasing α until it stagnates due to noise. One can also see that, for large α, $\hat{\mathbf{M}}$ computed via the pseudo-inverse is less accurate than directly extracting $\hat{\mathbf{M}}$ from (5.17) and imposing rank-1 structure. However, for small α, the opposite is true. Indeed, for decreasing α, the mixing vectors become less rank-1 like and our rank-1 model cannot attain a better estimate than the one given by Eckart–Young's theorem [80]. Also, note that the sources are estimated more accurately than the mixing vectors: the noise on the sources is more averaged out because this factor is much shorter in the decomposition ($K \ll I, J$) [60].

Figure 5.9 shows the relative errors for several choices of L_r. One can observe that for increasing L_r, the relative error decreases in the case of small α, i.e., in the case of little rank-1-like mixing vectors. On the other hand, little is lost through overmodeling (i.e., choosing L_r too large) for large α. In fact, we overmodel less than conventional methods as we exploit the low-rank structure. Hence, the choice of L_r is not so critical, see [51], [65]. In this case one also knows that the multilinear rank of $\mathcal{X}$ is bounded by $(\sum_{r=1}^{R} L_r, \sum_{r=1}^{R} L_r, R)$.

5.5.5 Compression versus accuracy

We investigate the trade-off between compression and accuracy which will lead to a better understanding on how to choose the segmentation parameters I_{n_m} and/or I_{n_s}. We do this by examining the accuracy of a low-rank approximation of various segmentations of a real-life EEG signal with a sample rate of 500 Hz. More precisely, we reshape the EEG signal of length

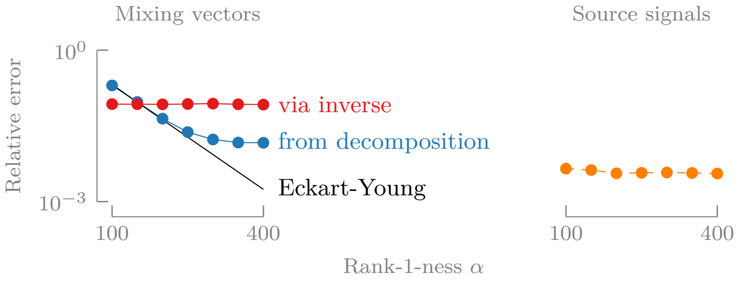


Figure 5.8: By computing the mixing vectors via the noisy observed data matrix and the pseudo-inverse of the estimated source matrix instead of extracting them directly from the decomposition, we can obtain a more accurate estimate of the mixing vectors when the rank-1 assumption does not hold entirely (i.e., for low α). Indeed, when the mixing vectors are less rank-1 like our model cannot attain a better estimate than the one given by Eckart–Young's theorem. *Here, we report the median across* 100 *experiments of the relative error on the mixing vectors, extracted from* (5.17) *and computed via the inverse of* $\hat{\mathbf{S}}$*, and the sources for varying rank-*1*-ness* α *and an SNR of* 15 *dB. The error bound given by the Eckart-Young theorem is shown in black.*

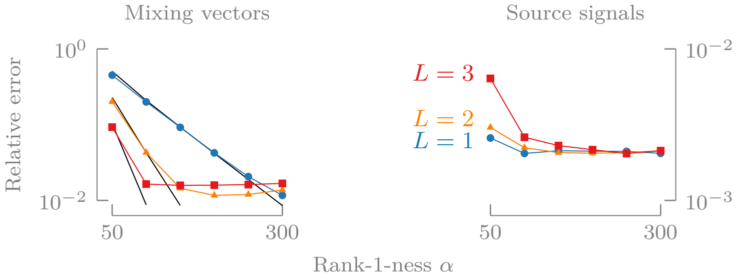


Figure 5.9: By increasing the rank of the model for the mixing vectors, one can decrease the relative error in the case of little rank-1-like mixing vectors (i.e., for low α). *Here, we report the median across* 100 *experiments of the relative error on the mixing vectors and the sources for varying rank-*1*-ness* α *of the mixing vectors and* 20 *dB SNR for* $L_1 = L_2 = L = 1$*,* $L_1 = L_2 = L = 2$*, and* $L_1 = L_2 = L = 3$*. The error estimate given by the Eckart-Young theorem is shown in black solid lines.*

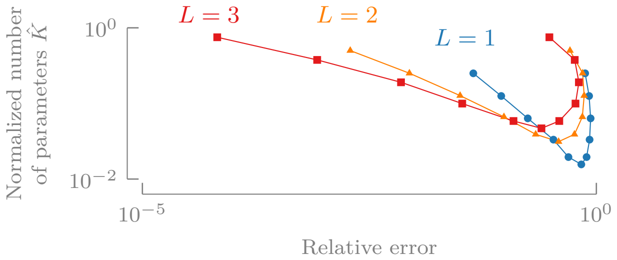


Figure 5.10: What is considered a *good* choice of parameters will depend on the needs in a particular application: there is a clear trade-off between the compression rate and the accuracy. *We report the normalized number of parameters $\hat{K}$ as a function of the relative error for a rank-1, -2, and -3 approximation of a segmented real-life EEG signal of length $K = 2^{14}$. The signal is reshaped into a $(I \times J)$ matrix with $I = 2^q$ and $J = 2^{14-q}$ such that $K = IJ$ with $q = 2, \ldots, 12$ and q increasing from left to right on the curve.*

$K = 2^{14}$ into a $(I \times J)$ matrix with $I = 2^q$ and vary $q = 2, \ldots, 12$, then $J = 2^{14-q}$ such that $K = IJ$. Subsequently, we approximate the reshaped signal with a rank-L model with $L = \{1, 2, 3\}$.

In Figure 5.10, we plot the normalized number of parameters $\hat{K} = L(I + J)/K$ versus the relative error ϵ of the rank-L approximation. We see a clear trade-off between compression and accuracy, hence, what is considered a “good” choice of parameters will depend on the needs in a particular application. First of all, the curves are not symmetric since segmentation is not symmetric in the modes that it creates. Note that one can easily improve the accuracy without affecting the compression rate by switching the values of I and J such that $I < J$ rather than $I > J$ for the same rank. For fixed I and J, increasing the rank can greatly improve the accuracy, e.g., when $I \ll J$ (left part of Figure 5.10). The original signal and two particular approximations are shown in Figure 5.11. Note the relative error decreased from 0.68 to 0.096 by taking $I < J$ and increasing L for the second approximation. On the other hand, the compression reduced from 96.88% to 86.72%.

In general, a good choice of the parameters will depend on the application. If compression is the objective, one should choose $I \approx J$ and L not too large. If, on the other hand, accuracy is the objective, one can try other choices of I and J and maybe a higher rank L. In practice, one can try a particular choice of parameters, perform a similar analysis as here on the estimated sources, and further refine the choice from there.

5.5.6 Consistency

We analyze the consistency of the mixing matrix estimate using our method by numerically checking the behavior for increasing sample size K. Consider

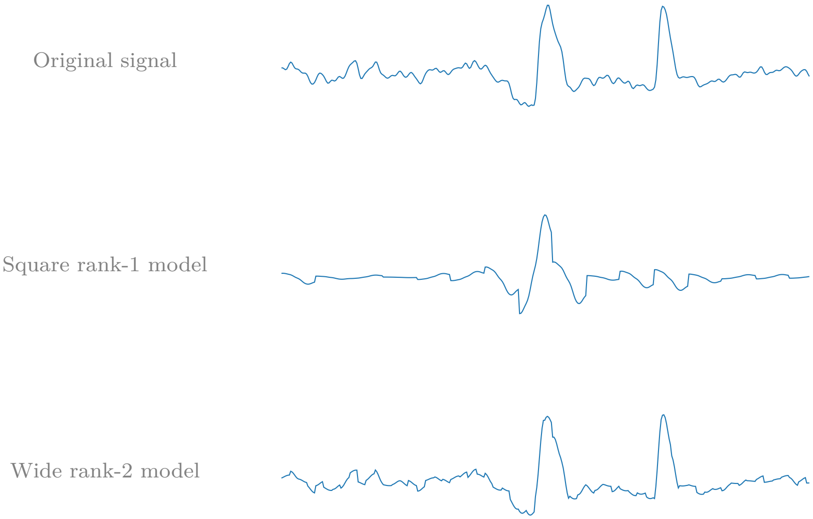


Figure 5.11: By making a *good* choice of parameters, one can obtain an accurate low-rank model, even for real-life signals. *This is illustrated for a real-life EEG signal and two approximations. The latter are obtained by first reshaping the original signal into a* $(2^7 \times 2^7)$ *(square) and* $(2^5 \times 2^9)$ *(wide) matrix, respectively, and then approximating them by a rank-*1 *and rank-*2 *matrix by truncating the singular value decomposition. The reconstructed signals are obtained by vectorizing these low-rank matrices. Only the first* 2000 *samples are shown. The rank-*2 *approximation is much better than the rank-*1 *as is also clear from Figure* 5.10.

a problem with $M = 10$ sensors and $R = 3$ sources of increasing length $K = \{10^3, 10^4, 10^5\}$. The mixing vectors are i.i.d. zero-mean unit-variance Gaussian vectors and the source signals are constructed as the vectorization of rank-1 ($L_1 = L_2 = 1$) matrices ($N = 2$) with zero-mean unit-variance Gaussian random columns and $I = J = \lfloor\sqrt{K}\rfloor$. For each value of K, we run 1000 experiments with 20 dB SNR and check the relative error on the estimate of the mixing matrix found from the decomposition in (5.15). By fitting a Gaussian distribution to the histogram of the relative errors, we obtain Figure 5.12. For increasing K, the estimates are getting more and more concentrated near zero relative error; note that there is no visible bias.

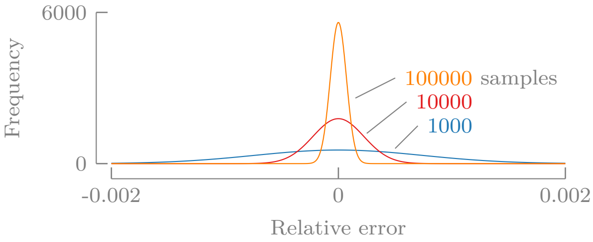


Figure 5.12: For increasing sample size, the relative errors on the estimates of the mixing matrix are getting more and more concentrated near zero.

5.5.7 Fetal electrocardiogram extraction

We use the method of section 5.2 for the extraction of the antepartum fetal electrocardiogram (FECG) from multilead cutaneous (i.e., recorded on the mother's skin) potential recordings. The FECG is important for analyzing the health and condition of the fetus. The elimination of the mother's dominant heartbeat in the ECG can be seen as a BSS problem and one can use methods such as ICA [55]. ICA, however, falls short when only a few samples or heartbeats are available. FECG extraction is not a large-scale problem, but it is useful to illustrate a few features of our approach. Our method is applicable here because the typical QRS[1] complexes in the ECG admit a low-rank approximation. In other words, we show that representability by a small number of parameters can be used as a ground for blind ECG signal separation. We illustrate our method for a real-life dataset.

The dataset contains eight observed signals, of which five abdominal and three thoracic; the dataset is available from DaISY[2]. Data acquisition and preprocessing is described in [32]. The sampling rate is 250 Hz. We only

---

[1] The QRS complex consists of the Q, R, and S wave that occur in rapid succession. The QRS complex represents the electrical impulse spreading through the ventricles, indicating ventricular depolarization [43].

[2] Available from `http://homes.esat.kuleuven.be/~smc/daisy/daisydata.html`

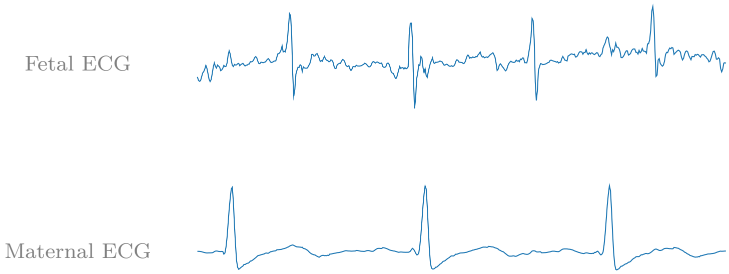


Figure 5.13: By exploiting the intrinstic low-rank structure of the QRS complexes in the ECG, our method can achieve a clear separation of the fetal and maternal ECG.

use the first 500 samples and scale each signal to unit norm. Each observed signal is segmented into a (25×20) matrix and the overall data set is stacked into a $(25 \times 20 \times 8)$ tensor. We use a rank-5 approximation for each source $(L_1 = L_2 = L_3 = L = 5)$. At least three sources are needed to extract the FECG; this is also the case for ICA [55]. We use this particular segmentation as to maximize the compression which is only an arbitrary choice. We determined L by a trial-and-error approach starting from a rank-10 approximation and then decreasing L. Little is lost by choosing a larger L anyway, see subsection 5.5.4. Figure 5.13 shows two recovered sources. One can verify that the heartbeats of the fetus are no longer visible in the ECG of the mother and vice versa, i.e, we have a clear separation. The frequency of the FECG is typically twice as high as the frequency of the maternal ECG (MECG), which can be observed as well.

5.5.8 Direction of arrival estimation

We use the method of section 5.4 for direction-of-arrival (DOA) estimation of signals impinging on a ULA. Applications include radar, sonar, wireless communications, and seismic exploration. Recently, there has been a trend towards large-scale array processing [130]. Our method is able to cope with a large number of sensors, where other methods fall short. We compare our results with two well-known DOA estimation methods, MUSIC and ESPRIT, in several scenarios [127].

Consider a ULA that consists of M uniformly spaced and omnidirectional antennas receiving signals from R narrow-band sources located in the far field. In that case, the problem can be described by (5.6) with the mixing vectors defined element-wise as $m_{mr} = \theta_r^{m-1}$ with $\theta_r = e^{-2\pi i \Delta \sin(\alpha_r) \lambda^{-1}}$. Δ is the inter-element spacing, the angle α_r to the normal is the rth DOA (i.e., $-90^\circ \leq \alpha_r \leq 90^\circ$), and λ denotes the wavelength. Note that the mixing vectors are Vandermonde vectors: $\mathbf{m}_r = \begin{bmatrix}1 & \theta_r & \theta_r^2 & \cdots & \theta_r^{M-1}\end{bmatrix}^{\mathrm{T}}$, hence, they admit a rank-1 representation [141], [166]. In a multipath setting, the mixing vectors are defined element-wise as $m_{mr} = \sum_{l=1}^{L_r} \theta_{lr}^{m-1}$ (ignoring path losses for simplicity), with L_r the number of paths for the rth source, admitting a low-rank representation. If the sources are located in the near field, the mixing vectors no longer admit a rank-1 representation but can still be well approximated by a low-rank model. If one also uses low-rank source models, we can use the method of section 5.4.

First, consider a ULA with $M = 64$ sensors and Δ equal to halve the wavelength. Although our method is applicable for a large number of sensors, we choose M rather small so we can compare with MUSIC and ESPRIT. The latter two methods have to compute a $M \times M$ covariance matrix and then apply an eigenvalue decomposition (EVD). These steps can be computationally expensive because they have a complexity of $\mathcal{O}(M^2K)$ and $\mathcal{O}(M^3)$ (or $\mathcal{O}(M^2)$ when using iterative methods [190])[3], respectively, rendering such methods infeasible for large M and K. Moreover, MUSIC has to evaluate the MUSIC spectrum for many angles in order to estimate the DOAs accurately. Here, we evaluated the MUSIC spectrum in 10^4 equidistant angles in $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Note that the number of evaluation points bounds the attainable accuracy. Consider $R = 2$ low-rank sources: $s_r(t) = \sin(10\pi r t)$ with $K = 1024$ equidistant samples in $[0, 1]$. The sources are in line-of-sight and impinge on the ULA with $\alpha_{11} = 32^\circ$ and $\alpha_{12} = 34^\circ$. We use a second-order ($N_s = 2$) rank-2 ($P_1 = P_2 = 2$) approximation for both sources with $J_1 = J_2 = 32$ and a second-order ($N_m = 2$) rank-1 ($L_1 = L_2 = 1$) approximation for both mixing vectors with $I_1 = I_2 = 8$. Note that the model of the sources and mixing vectors requires only 126 and 15 values instead of 1024 and 64, respectively, see Table 5.2. This results in a compression of $1 - P_r\frac{J_1+J_2-N_s+1}{K} = 87.70\%$ and $1 - L_r\frac{I_1+I_2-N_m+1}{M} = 76.56\%$, respectively. In Figure 5.14 (left), we report the median of the relative errors on the DOAs ϵ_α. It is clear that the dedicated methods estimate the DOAs more accurately than our method. On the other hand, by exploiting the low-rank structure, we show that it is still possible to get fairly accurate estimates in comparison with well-known dedicated methods. Moreover, our method is applicable for large M.

In a second experiment, we add a third source ($R = 3$) that impinges on the ULA from two different paths ($L_3 = 2$): $\alpha_{13} = -15^\circ$ and $\alpha_{23} = 67^\circ$. We use a third-order ($N_m = 3$) rank-1 and rank-2 approximation for the first two and

---

[3] Consider an $M \times K$ data matrix with $M = I^N$ and $K = I$, which can be segmentized in an $(N + 1)$th-order tensor with size $I \times I \times \cdots \times I$. While the computation of the $M \times M$ covariance matrix and the EVD in MUSIC and ESPRIT algorithms have a computational complexity of $\mathcal{O}(I^{2N+1})$ and $\mathcal{O}(I^{3N})$ (or, $\mathcal{O}(I^{2N})$), resp., the per-iteration computational complexity of a Gauss-Newton algorithm with dogleg trust regions for the CPD is only $\mathcal{O}(2(N+1+\mathrm{it}_{\mathrm{TR}})RI^{N+1} + \frac{8}{3}(N+1)^3R^3I^3)$ with $\mathrm{it}_{\mathrm{TR}}$ the number of trust-region iterations [180].

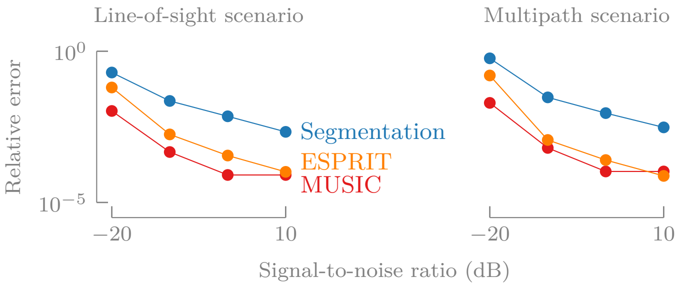


Figure 5.14: Eventhough dedicated methods for direction-of-arrival estimation such as MUSIC and ESPRIT are more accurate than our method, we still obtain fairly accurate estimates using our general framework for both the line-of-sight and multipath scenarios. *We report the median across* 100 *experiments of the relative error on the direction-of-arrival angles as a function of the signal-to-noise ratio for the line-of-sight and multipath* far-field *scenario using segmentation, ESPRIT, and MUSIC.*

last mixing vector, respectively, with $I_1 = I_2 = I_3 = 4$. We choose $N_m > 2$ such that the different DOAs of the third source can be found directly from the estimated vectors $\mathbf{u}_{13}^{(1)}$ and $\mathbf{u}_{23}^{(1)}$ (instead of the column space of $\mathbf{S}_3$), see the discussion of uniqueness in subsection 5.2.3. Note that one simply has to increase the rank L_r in order to cope with a multipath source, while MUSIC and ESPRIT need additional spatial smoothing [164]. The results are shown in Figure 5.14 (right).

Next, we use the same setup as in the first experiment but with two near-field sources defined by a DOA and range relative to the first antenna: $\alpha_1 = -17°$, $w_1 = 2(M-1)\Delta$, $\alpha_2 = 41°$, and $w_2 = 3(M-1)\Delta$. We compare our results with a two-dimensional version of MUSIC [110]. Figure 5.15 shows the median of the relative errors on the DOAs ϵ_α and the ranges ϵ_w. MUSIC estimates both the DOA and range more accurately but is even more computationally expensive because now one has to evaluate a two-dimensional spectrum for many angles and ranges. Here, we used 10^2 equidistant angles and ranges in $[-\frac{\pi}{2}, \frac{\pi}{2}]$ and $[5, 12]$, respectively. In order to cope with near-field sources in our approach, one simply has to increase the rank L_r.

In the final experiment, we use again the same setup as the first experiment but now with $M = 9$ and $K = 100$ with $J_1 = J_2 = 10$ and $I_1 = I_2 = 3$. As can be seen from Figure 5.16, MUSIC fails to distinguish close DOAs when only a few samples are available and the SNR is low [127]. A small number of sensors M flattens the peaks in the MUSIC spectrum, making the problem more difficult. Our method can still estimate the DOAs accurately in such a setup because it is deterministic, performing even better than ESPRIT.

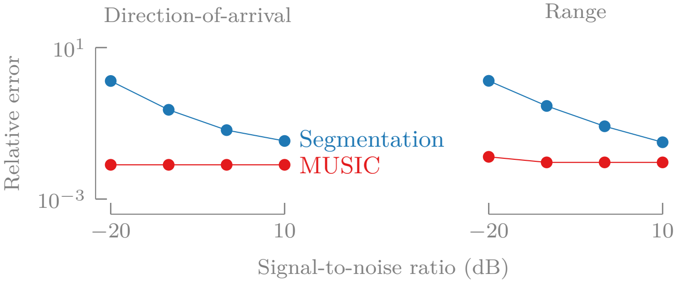


Figure 5.15: Eventhough our segmentation-based method is less accurate than the dedicated MUSIC method, we simply have to increase the rank in order to cope with *near-field* sources. *We report the median across* 100 *experiments of the relative error on the direction-of-arrival angles and ranges as a function of the signal-to-noise ratio for the* near-field *scenario using segmentation and MUSIC.*

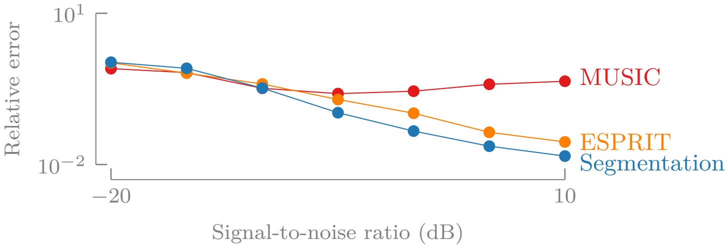


Figure 5.16: In contrast to MUSIC, our method can distinguish direction-of-arrival angles that are close when only a few samples are available and the signal-to-noise ratio is low. *We report the median across* 100 *experiments of the relative error on the direction-of-arrival angles as a function of the signal-to-noise ratio for the small-scale line-of-sight experiment using segmentation, ESPRIT, and MUSIC.*

5.6 Conclusion

In this chapter, we have introduced a new method for BSS that exploits the fact that many real-life signals are compressible. We expressed this by assuming that the tensorized sources can be well approximated by a low-rank model. In other words, we assume that the sources can be well approximated by sums of Kronecker products of smaller vectors. As such, we have demonstrated that, if the sources indeed admit such a low-rank representation/approximation, the BSS problem boils down to the computation of a decomposition of the resulting tensorized observed data matrix. It is precisely the compressibility, which is essential in large-scale problems, that makes it very likely that the tensor decomposition is unique. Hence, our method provides a unique solution to the BSS problem and a way to cope with large-scale problems. Furthermore, we applied the same strategy to the mixing level motivated by an increasing number of sensors and sensor density in fields such as biomedical sciences and array processing. Moreover, combining both strategies simultaneously allowed the exploitation of low-rank structure on both levels of the BSS problem. We have illustrated our methods with two applications: FECG and DOA estimation for large-scale ULAs. We note that it is possible to impose constraints on the sources and/or mixture when applicable, e.g., statistical independence of the sources as in ICA. Such variants are out of the scope of this chapter. Although we focused on the CPD for modeling the tensorized sources and/or mixture, it is possible to consider other tensor models such as tensor trains (TTs) and hierarchical Tucker [96]. The latter are often used in tensor-based scientific computing because they combine large compression rates with good numerical properties. For the CPD of very large tensors, algorithms such as the ones in [171], [209], [214] can be used.

Tensor-based large-scale blind system identification using segmentation

6

ABSTRACT | Many real-life signals can be described in terms of much fewer parameters than the actual number of samples. Such compressible signals can often be represented very compactly with low-rank matrix and tensor models. The authors have adopted this strategy to enable large-scale instantaneous blind source separation. In this chapter, we generalize the approach to the blind identification of large-scale convolutive systems. In particular we apply the same idea to the system coefficients of finite impulse response systems. This allows us to reformulate blind system identification as a structured tensor decomposition. The tensor is obtained by applying a deterministic tensorization technique called segmentation on the observed output data. Exploiting the low-rank structure of the system coefficients enables a unique identification of the system and estimation of the inputs. We obtain a new type of deterministic uniqueness conditions. Moreover, the compactness of the low-rank models allows one to solve large-scale problems. We illustrate our method for direction-of-arrival estimation in large-scale antenna arrays and neural spike sorting in high-density microelectrode arrays.

This chapter is a slightly adapted version of M. Boussé, O. Debals, and L. De Lathauwer, "Tensor-based large-scale blind system identification using segmentation", *IEEE Transactions on Signal Processing*, vol. 65, no. 21, pp. 5770–5784, Nov. 2017. The figures have been updated for consistency.

6.1 Introduction

In blind system identification (BSI) one wishes to identify an unknown system using *only* the measured output values [1]. In this chapter, we specifically limit ourselves to the blind identification of finite impulse response (FIR) systems. Hence, the outputs are convolutive mixtures of the inputs in contrast with instantaneous blind source separation (BSS) [45]. Also, we define the goal of BSI to be both the estimation of the system coefficients and the reconstruction of the inputs; we do not make a distinction. In order to make the BSI problem feasible, additional assumptions have to be imposed on the inputs or the system coefficients. The choice of a particular assumption typically depends on the application; examples are independent inputs, finite alphabet, and constant modulus [1]. BSI is an important problem with a variety of applications in (biomedical) signal processing, image processing, and sensor array processing [107], [129], [206].

Recently, there is a trend to more sensors and larger sensor density in several domains. Biomedical examples include high-density surface electromyogram (sEMG) and wireless body area networks (WBANs) based on electroencephalography (EEG) and electrocorticography (ECoG) [13], [107], [161]. BSS and BSI are typical problems in these applications [21]. For example, the separation of action potentials of the muscle's motor units in sEMG recordings is typically modeled using BSI [107]. In array processing and telecommunications, an increase in the number of antennas is seen, known as massive MIMO [130]. Here, BSI using FIR models can be used to determine the direction-of-arrivals (DOAs) of narrow-band signals impinging on uniform linear arrays (ULAs) and rectangular arrays (URAs) from the far field [127].

The key idea to tackle such large-scale problems is known from compressive sensing: there is often an excessive number of entries compared to the actual amount of information contained in the system coefficients [34]. In other words, there is some structure and/or sparsity in the system coefficients that allows one to model it much more compactly [190]. Such signals are called compressible and they can typically be represented by parsimonious models such as low-rank higher-order tensor models. This approach is known from tensor-based scientific computing in high dimensions [96], [214]. The compactness of these models, especially in the case of higher-order tensors, has allowed one to solve problems in a number of unknowns that exceeds the number of atoms in the universe. The authors have adopted this particular strategy to enable large-scale BSS [18], [21]. In this chapter, we extend the strategy to the system coefficients in convolutive BSI.

The proposed method tensorizes the measured output values using a particular tensorization technique called segmentation [21], [62]. We show that large-scale convolutive BSI reduces to a structured decomposition of the resulting tensor. In general, the decomposition is a generalization of a particular block term decomposition (BTD) [49] called a flower decomposition that was first introduced in [18], [21]. The latter has a block-Toeplitz structure in this case due to the convolutive nature of the FIR model that is used. The above approach allows us to exploit the underlying compactness of the system coefficients using low-rank models, enabling a unique identification of both the system and the inputs of large-scale BSI problems. Segmentation can be interpreted as a compact version of Hankel-based tensorization [51]. As such, one can show that the approach is exact for system coefficients that can be modeled as exponential polynomials but also a much broader class of signals [21]. Moreover, one can show that our method works well for system coefficients that admit a good polynomial approximation.

To best of the authors' knowledge, our segmentation-based method is the first method for (very) large-scale BSI, using a similar philosophy as in tensor-based scientific computing. The contributions of this chapter include a discussion of the uniqueness conditions for the flower decomposition and a new algebraic method to compute it. Also, we provide novel uniqueness conditions for the BSI problem with and without exploiting the block-Toeplitz structure. Furthermore, we prove a new result for the low-rank approximation of periodic signals of which the period may have been estimated inaccurately. Additionally, we perform a parameter analysis and especially investigate the influence of the FIR system order and the low-rank model parameters. Finally, our method allows to accurately estimate the direction-of-arrivals (DOAs) in large-scale URAs. Moreover, it enables DOA estimation in non-uniform arrays and can handle broken antennas. We also illustrate our method for spike sorting in high-density microelectrode arrays. First results of our approach were briefly discussed in [19].

We conclude this section with an overview of the notation and definitions. Next, we discuss the flower decomposition in section 6.2. We reformulate convolutive BSI as a flower decomposition using segmentation in section 6.3. Simulations and applications are presented in section 6.4 and section 6.5.

6.1.1 Notation and definitions

Vectors and matrices are denoted by bold lowercase and bold uppercase letters, e.g., $\mathbf{a}$ and $\mathbf{A}$, respectively. Tensors are a higher-order generalization of the former and are denoted by calligraphic letters, e.g., $\mathcal{A}$. We denote index upper bounds by italic capitals, e.g., $1 \leq i \leq I$. The $(i_1, i_2, \ldots, i_N)$th entry of an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ (with $\mathbb{K}$ meaning $\mathbb{R}$ or $\mathbb{C}$) is denoted by $a_{i_1 i_2 \ldots i_N}$. An element of a sequence is indicated by a superscript between parentheses, e.g., the nth matrix $\mathbf{A}^{(n)}$. The matrix transpose is indicated by $\bullet^{\mathrm{T}}$. The unit vector $\mathbf{e}_i$ has a one in the ith row. The $I \times I$ identity matrix is denoted by $\mathbf{I}_I$. The entries of the nth compound matrix of $\mathbf{A} \in \mathbb{K}^{I \times J}$, denoted by $C_n(\mathbf{A}) \in \mathbb{K}^{\binom{I}{n} \times \binom{J}{n}}$, equal the $n \times n$ minors of $\mathbf{A}$ ordered lexicographically.

The rows and columns of a matrix can be generalized for higher-order tensors to mode-n vectors, which are defined by fixing every index except the nth. A mode-n matrix unfolding of $\mathcal{A}$ is a matrix $\mathbf{A}_{(n)}$ with the mode-n vectors as its columns following the ordering convention in [125]. Vectorization of $\mathcal{A}$, denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \cdots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1 + \sum_{k=1}^{N} (i_k - 1) J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$. The kth frontal slice $\mathbf{X}_k$ of a third-order tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$ is obtained by fixing only the last index. We denote the outer and Kronecker product as $\otimes$ and $\otimes$, respectively. They are related through a vectorization: $\operatorname{vec}(\mathbf{a} \otimes \mathbf{b}) = \mathbf{b} \otimes \mathbf{a}$. We denote the Khatri–Rao product as $\odot$.

6.1.2 Tensor decompositions

The rank of a tensor equals the minimal number of rank-1 tensors that generate the tensor as their sum. A rank-1 tensor is defined as the outer product of nonzero vectors. The rank of a mode-n unfolding of a tensor is the mode-n rank. The multilinear rank is defined as the tuple of these mode-n ranks.

Definition 26. A *polyadic decomposition* (PD) writes an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ as a sum of R rank-1 terms:

$$\mathcal{A} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(N)}. \tag{6.1}$$

The columns of the factor matrices $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$ are equal to the factor vectors $\mathbf{u}_r^{(n)}$ for $1 \leq r \leq R$. The PD is called *canonical* (CPD) when R is equal to the rank of $\mathcal{A}$. The mode-n matrix unfolding of the PD defined in (6.1) is given by:

$$\mathbf{A}_{(n)} = \mathbf{U}^{(n)} (\mathbf{U}^{(N)} \odot \cdots \odot \mathbf{U}^{(n+1)} \odot \mathbf{U}^{(n-1)} \odot \cdots \odot \mathbf{U}^{(1)})^{\mathrm{T}}.$$

The CPD is *essentially unique* if it is unique up to trivial permutation of the rank-1 terms and scaling and counter-scaling of the factors in the same term. The decomposition is unique under rather mild conditions which is a powerful advantage of tensors over matrices in many applications. See [70]–[73], [75] and references therein for state-of-the-art uniqueness conditions. The CPD has been used in many applications within signal processing, biomedical sciences, data mining and machine learning, see [41], [125], [168].

Definition 27. A *block term decomposition* (BTD) of a third-order tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$ *in multilinear rank-*$(P_r, P_r, 1)$ *terms* for $1 \leq r \leq R$ is a decomposition of the form:

$$\mathcal{X} = \sum_{r=1}^{R} (\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}) \otimes \mathbf{c}_r = \sum_{r=1}^{R} \left(\sum_{p=1}^{P_r} \mathbf{a}_{pr} \otimes \mathbf{b}_{pr} \right) \otimes \mathbf{c}_r, \tag{6.2}$$

in which $\mathbf{A}_r \in \mathbb{K}^{I \times P_r}$ and $\mathbf{B}_r \in \mathbb{K}^{J \times P_r}$ have full column rank P_r and $\mathbf{c}_r$ is nonzero. Also, we define $R' = \sum_{r=1}^{R} P_r$. The mode-3 unfolding $\mathbf{X}_{(3)} \in \mathbb{K}^{K \times IJ}$ of (6.2) is given by

$$\mathbf{X}_{(3)} = \mathbf{C}\left[\operatorname{vec}\left(\mathbf{A}_1\mathbf{B}_1^{\mathrm{T}}\right) \ \cdots \ \operatorname{vec}\left(\mathbf{A}_R\mathbf{B}_R^{\mathrm{T}}\right)\right]^{\mathrm{T}}.$$

The decomposition in (6.2) can be interpreted as a CPD with proportional columns in the last factor matrix. Define the following factor matrices $\mathbf{A} = \begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2 & \cdots & \mathbf{A}_R\end{bmatrix} \in \mathbb{K}^{I \times R'}$, $\mathbf{B} = \begin{bmatrix}\mathbf{B}_1 & \mathbf{B}_2 & \cdots & \mathbf{B}_R\end{bmatrix} \in \mathbb{K}^{J \times R'}$, and $\mathbf{C}^{(\mathrm{ext})} = \begin{bmatrix}\mathbf{1}_{P_1}^{\mathrm{T}} \otimes \mathbf{c}_1 & \cdots & \mathbf{1}_{P_R}^{\mathrm{T}} \otimes \mathbf{c}_R\end{bmatrix} \in \mathbb{K}^{K \times R'}$. As such, we have a rank-R' CPD with the following mode-3 unfolding:

$$\mathbf{X}_{(3)} = \mathbf{C}^{(\mathrm{ext})}(\mathbf{B} \odot \mathbf{A})^{\mathrm{T}}. \tag{6.3}$$

The BTD is *essentially* unique if it is unique up to trivial permutation of the rth and r'th term, if $P_r = P_{r'}$, and scaling and counter-scaling of $(\mathbf{A}_r\mathbf{B}_r^{\mathrm{T}})$ and $\mathbf{c}_r$ in the same term. We repeat a uniqueness result for this particular decomposition that will be used later in section 6.3 [49, Theorem 4.1]:

Theorem 3. Consider a BTD in multilinear rank-$(P_r, P_r, 1)$ terms of $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$ as in (6.2) with $I, J \geq R'$. The decomposition is essentially unique if $\mathbf{A} = \begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2 & \cdots & \mathbf{A}_R\end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix}\mathbf{B}_1 & \mathbf{B}_2 & \cdots & \mathbf{B}_R\end{bmatrix}$ have full column rank and $\mathbf{C} = \begin{bmatrix}\mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_{\mathbf{r}}\end{bmatrix}$ does not have proportional columns.

The BTD allows one to model more complex phenomena because of the more general block terms in comparison to the rank-1 terms of the CPD in (6.1) [51]–[53]. Other types of BTDs and uniqueness conditions can be found in [49], [51].

6.2 Decomposition in (rank-$P_r \otimes$ vector) terms

In this chapter, we introduce a new method for convolutive BSI. We show in section 6.3 that our method reformulates BSI as a (structured) decomposition of a higher-order tensor in (rank-$P_r \otimes$ vector) terms. This decomposition was first introduced in [18], [21] and is also called the *flower decomposition*. One can see the rank-P_r part as the petals and the vector as the stem of a flower, see Figure 6.1. The decomposition can be interpreted as a higher-order generalization of the BTD in multilinear rank-$(P_r, P_r, 1)$ terms as defined in subsection 6.1.2. In subsection 6.2.1 and subsection 6.2.2 we define the decomposition and discuss uniqueness properties, respectively. We propose a new algebraic method for its computation in subsection 6.2.3.

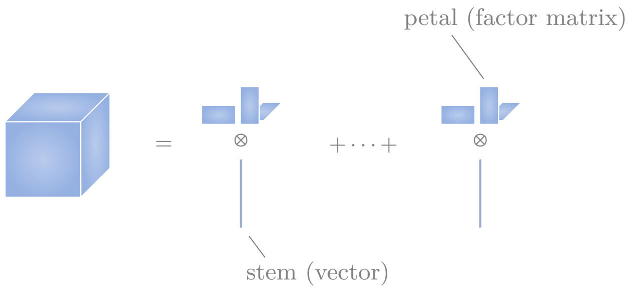


Figure 6.1: Decomposition of a fourth-order tensor $\mathcal{X}$ (illustrated as a third-order tensor) in (rank-P_r ⊗ vector) terms. One can see each factor matrix of the rank-P_r tensor as a petal of the flower and the vector as the stem, hence, “flower” decomposition.

6.2.1 Definition

We generalize the BTD in multilinear rank-$(P_r, P_r, 1)$ terms of a third-order tensor from subsection 6.1.2. The decomposition is called a decomposition in (rank-P_r ⊗ vector) [18], [21].

Definition 28. A *flower decomposition* is a decomposition of an $(N+1)$th-order tensor $\mathcal{X} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N \times K}$ *in (rank-P_r ⊗ vector) terms* for $1 \leq r \leq R$ of the form:

$$\mathcal{X} = \sum_{r=1}^{R} \left(\sum_{p=1}^{P_r} \mathbf{u}_{pr}^{(1)} \otimes \mathbf{u}_{pr}^{(2)} \otimes \cdots \otimes \mathbf{u}_{pr}^{(N)} \right) \otimes \mathbf{s}_r \tag{6.4}$$

with factor matrices $\mathbf{U}^{(n)} = \left[\mathbf{U}_1^{(n)} \; \mathbf{U}_2^{(n)} \cdots \mathbf{U}_R^{(n)} \right] \in \mathbb{K}^{I_n \times R'}$ and $\mathbf{S} \in \mathbb{K}^{K \times R}$ in which $\mathbf{U}_r^{(n)} = \left[\mathbf{u}_{1r}^{(n)} \; \mathbf{u}_{2r}^{(n)} \cdots \mathbf{u}_{P_r r}^{(n)} \right] \in \mathbb{K}^{I_n \times P_r}$ and $R' = \sum_{r=1}^{R} P_r$.

Note that each term is an outer product of a rank-P_r tensor and a vector. Hence, it is clear that this decomposition boils down to a BTD in multilinear rank-$(P_r, P_r, 1)$ terms for third-order tensors, i.e., when $N = 2$. In that case, however, the factor matrices $\mathbf{A}_r$ and $\mathbf{B}_r$ are not unique without additional assumptions (but their products are) because $\mathbf{A}_r {\mathbf{B}_r}^{\mathrm{T}} = (\mathbf{A}_r \mathbf{D}_r^{-1})(\mathbf{B}_r {\mathbf{D}_r}^{\mathrm{T}})^{\mathrm{T}}$ for any square nonsingular matrix $\mathbf{D}_r$. This is not the case for $N > 2$ because essential uniqueness is guaranteed under mild conditions, see subsection 6.1.2. Finally, if $P_r = 1, 1 \leq r \leq R$, then (6.4) reduces to a PD as defined in (6.1).

6.2.2 Uniqueness

Uniqueness conditions for a decomposition of an $(N+1)$th-order tensor in (rank-P_r ⊗ vector) terms can be obtained by reworking the decomposition into a set of coupled BTDs in rank-$(P_r, P_r, 1)$ terms and assuming the common factor matrix has full column rank [184]. This is possible by keeping one factor matrix in a common mode and combining the remaining modes in the first and second mode while ignoring the Khatri–Rao structure. Doing this for N possible combinations, leads to a coupled decomposition equivalent to the original one, as we will explain here. The former is unique up to trivial permutation of the coupled multilinear rank-$(P_r, P_r, 1)$ terms as well as scaling and counterscaling of the matrices and vectors within the same term. We call the coupled decomposition essentially unique when it is only subject to these indeterminacies.

Consider a tensor $\mathcal{X} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N \times K}$ that admits a flower decomposition with mode-$(N+1)$ unfolding given by

$$\mathbf{X}_{(N+1)} = \mathbf{S} \left(\operatorname{vec}\left(\sum_{p=1}^{P_r} \mathbf{u}_{p1}^{(1)} \otimes \mathbf{u}_{p1}^{(2)} \otimes \cdots \otimes \mathbf{u}_{p1}^{(N)} \right) \cdots \right.$$
$$\left. \operatorname{vec}\left(\sum_{p=1}^{P_r} \mathbf{u}_{pR}^{(1)} \otimes \mathbf{u}_{pR}^{(2)} \otimes \cdots \otimes \mathbf{u}_{pR}^{(N)} \right) \right)^{\mathrm{T}},$$

or equivalently,

$$\mathbf{X}_{(N+1)} = \mathbf{S}^{(\mathrm{ext})} \left(\mathbf{U}^{(N)} \odot \mathbf{U}^{(N-1)} \odot \cdots \odot \mathbf{U}^{(2)} \odot \mathbf{U}^{(1)} \right)^{\mathrm{T}} \tag{6.5}$$

with $\mathbf{S}^{(\mathrm{ext})} = \left[\mathbf{1}_{P_1}^{\mathrm{T}} \otimes \mathbf{s}_1 \ \cdots \ \mathbf{1}_{P_R}^{\mathrm{T}} \otimes \mathbf{s}_R \right] \in \mathbb{K}^{K \times R'}$. Consider N different partitionings of the N factor matrices $\mathbf{U}^{(N)}$ into two sets. The factor matrices in each set can be collected in factor matrices $\mathbf{A}^{(w)}$ and $\mathbf{B}^{(w)}$. As such, we obtain several matrix representations of the tensor $\mathcal{X}$ of the form:

$$\mathbf{X}^{(w)} = \mathbf{S}^{(\mathrm{ext})} \left(\mathbf{B}^{(w)} \odot \mathbf{A}^{(w)} \right)^{\mathrm{T}} \text{ for } 1 \leq w \leq N \tag{6.6}$$

with $\mathbf{S}^{(\mathrm{ext})}$ acting as a common factor for all N possibilities. Clearly, every decomposition in (6.6) is a mode-3 unfolding of a BTD in rank-$(P_r, P_r, 1)$ terms, see (6.3). Mathematically, we have that $\mathbf{X}^{(w)} \in \mathbb{K}^{K \times I'}$, $\mathbf{A}^{(w)} = \bigodot_{\gamma \in \Gamma_w} \mathbf{U}^{(\gamma)} \in \mathbb{K}^{I'_w \times R'}$ and $\mathbf{B}^{(w)} = \bigodot_{\upsilon \in \Upsilon_w} \mathbf{U}^{(\upsilon)} \in \mathbb{K}^{J'_w \times R'}$ with $I'_w = \prod_{\gamma \in \Gamma_w} I_\gamma$, $J'_w = \prod_{\upsilon \in \Upsilon_w} I_\upsilon$, and $I' = \prod_{n=1}^{N} I_n$. The sets Γ_w and Υ_w satisfy $\Gamma_w \ \cup \Upsilon_w = \{1, \ldots, N\}$ and $\Gamma_w \cap \Upsilon_w = \emptyset$.

The Khatri-Rao products in $\mathbf{A}^{(w)}$ and $\mathbf{B}^{(w)}$ are ignored. Nevertheless, the matrix representation in (6.5) and the coupled decomposition represented in (6.6) are equivalent [184]. Hence, the full decomposition in (rank-$P_r \otimes$ vector) terms of the $(N+1)$th-order tensor $\mathcal{X}$ corresponds to a coupled decomposition in rank-$(P_r, P_r, 1)$ terms of third-order tensors in which part of the structure has been ignored. As such, the uniqueness results derived in [184] can be used. For example, if one of the BTDs is unique and $\mathbf{S}$ has full column rank, then the (rank-$L_r \otimes$ vector) decomposition is unique. This example is in fact trivial; the uniqueness results in [184] go further than that.

Let us illustrate the approach for a 4th-order tensor $\mathcal{X} \in \mathbb{K}^{I_1 \times I_2 \times I_3 \times K}$ that admits the following decomposition

$$\mathcal{X} = \sum_{r=1}^{R} \left(\sum_{p=1}^{P_r} \mathbf{u}_{pr}^{(1)} \otimes \mathbf{u}_{pr}^{(2)} \otimes \mathbf{u}_{pr}^{(3)} \right) \otimes \mathbf{s}_r .$$

This decomposition can be written as three decompositions in multilinear rank-$(P_r, P_r, 1)$ terms that are coupled via the factor matrix in the fourth mode $\mathbf{S}^{(\text{ext})}$. Hence, one obtains

$$\begin{cases} \mathbf{X}^{(1)} = \left(\left(\mathbf{U}^{(1)} \odot \mathbf{U}^{(2)}\right) \odot \mathbf{U}^{(3)}\right) \mathbf{S}^{(\text{ext})\text{T}}, \\ \mathbf{X}^{(2)} = \left(\left(\mathbf{U}^{(1)} \odot \mathbf{U}^{(3)}\right) \odot \mathbf{U}^{(2)}\right) \mathbf{S}^{(\text{ext})\text{T}}, \\ \mathbf{X}^{(3)} = \left(\left(\mathbf{U}^{(2)} \odot \mathbf{U}^{(3)}\right) \odot \mathbf{U}^{(1)}\right) \mathbf{S}^{(\text{ext})\text{T}}. \end{cases}$$

The matrices $\mathbf{U}^{(n)}$, $1 \leq n \leq 3$ are combined in the first and second mode in three different ways. Ignoring the Khatri-Rao structure in the first mode, the coupled decomposition of third-order tensors $\mathbf{X}^{(n)}$, $1 \leq n \leq 3$, is equivalent with the decomposition of the fourth-order tensor $\mathcal{X}$.

6.2.3 Algebraic method

Often, algebraic methods for computing a tensor decomposition provide a good initialization for optimization-based methods. Here, we present an algebraic method for the flower decomposition defined in (6.4). We do this by generalizing an algebraic method for computing a BTD in multilinear rank-$(P_r, P_r, 1)$ terms that was proposed in [49]. This method assumes that the BTD satisfies Theorem 3 and reduces the computation to the computation of a generalized eigenvalue decomposition (GEVD). The algorithm is available in Tensorlab as `ll1_gevd` [215].

A BTD in multilinear rank-$(P_r, P_r, 1)$ terms can be interpreted as a CPD with proportional columns in the third factor matrix as explained in subsection 6.1.2. As such, it can be shown that the algebraic method of [49] boils down to the following three steps. First, we compute a solution of (6.2) using an algebraic method for a rank-R' CPD such as `cpd_gevd` from Tensorlab obtaining $\mathbf{C}^{(\text{ext})}$. Next, we cluster the R' columns of $\mathbf{C}^{(\text{ext})}$ into R clusters of size P_r. We use the k-lines method for clustering in order to accommodate for scaling and sign invariance. The rth cluster center then serves as an estimate for the rth column of $\mathbf{C}$. Finally, we compute the factor matrices $\mathbf{A}_r$ and $\mathbf{B}_r$ for $1 \leq r \leq R$ by reshaping the rth column of $\left(\mathbf{C}^{\dagger}\mathbf{X}_{(3)}\right)^{\text{T}} = \left[(\mathbf{B}_1 \odot \mathbf{A}_1)\mathbf{1}_{P_1} \quad \cdots \quad (\mathbf{B}_R \odot \mathbf{A}_R)\mathbf{1}_{P_R}\right]$ into a $(I \times J)$ matrix and computing a rank-P_r approximation of this matrix. This approach can be generalized to the flower decomposition as it can be interpreted as a higher-order generalization of the BTD in multilinear rank-$(P_r, P_r, 1)$ terms. Hence, we also interpret the decomposition as a CPD with proportional columns in the last factor matrix and apply the same scheme as above. The resulting method is called `lvec_gevd`, and is outlined in Algorithm 6.1.

Algorithm 6.1: Algebraic method for a decomposition of an $(N+1)$th-order tensor $\mathcal{X}$ in (rank-L_r ⊗ vector) terms.

```
**Input:** $\mathcal{X}$, $R$, $R'$, and $\{P_r\}_{r=1}^{R}$
**Output:** $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ and $\mathbf{S}$
Compute a CPD of $\mathcal{X}$ with $R'$ terms using a GEVD obtaining $\mathbf{S}^{(\mathrm{ext})}$
Cluster the $R'$ columns of $\mathbf{S}^{(\mathrm{ext})}$ into $R$ clusters. Use the cluster centers as an estimate for $\mathbf{S}$
Obtain the $r$th factor matrix $\mathbf{U}_r^{(n)}$ for $1 \leq n \leq N$ by reshaping the $r$th column of $\left(\mathbf{S}^{\dagger}\mathbf{X}_{(N+1)}\right)^{\mathrm{T}} = \left[(\mathbf{U}_1^{(N)} \odot \cdots \odot \mathbf{U}_1^{(1)})\mathbf{1}_{P_1} \cdots (\mathbf{U}_R^{(N)} \odot \cdots \odot \mathbf{U}_R^{(1)})\mathbf{1}_{P_R}\right]$ into an $(I_1 \times I_2 \times \cdots \times I_N)$ tensor and computing a rank-$P_r$ approximation of this tensor algebraically.
```

6.3 Large-scale BSI using segmentation

In large-scale applications, signals and systems often admit a compact representation. In this section we present a new method for large-scale convolutive BSI that exploits this, by reformulating the problem as a block-Toeplitz structured flower decomposition. We show that this approach allows one to uniquely identify both the coefficients and the inputs of large-scale systems. We define the BSI problem in subsection 6.3.1. Next, we motivate the working hypothesis of low-rank system coefficients in subsection 6.3.2 and derive our method in subsection 6.3.3. We also consider uniqueness properties in subsection 6.3.4. Finally, we investigate the block-Toeplitz structure of the decomposition in subsection 6.3.5.

6.3.1 Blind system identification

The goal of convolutive blind system identification (BSI) is to identify the coefficients of the system and/or the inputs using only the output data. More specifically, we consider discrete linear time-invariant systems with M outputs, R inputs, and system order L. The mth output of the finite impulse response (FIR) system is described by:

$$x_m[k] = \sum_{r=1}^{R}\sum_{l=0}^{L} g_{mr}[l]s_r[k-l] + n_m[k], \ 1 \leq k \leq K. \tag{6.7}$$

The FIR coefficients from the rth input to the mth output are denoted by $g_{mr}[l]$ for $0 \leq l \leq L$. The rth input is denoted as $s_r[k]$ and the additive noise on the mth output as $n_m[k]$. Equation (6.7) can be expressed in matrix form:

$$\mathbf{X} = \sum_{l=0}^{L} \mathbf{G}^{(l)} \mathbf{S}^{(l)^{\mathrm{T}}} = \mathbf{G}\mathbf{S}^{\mathrm{T}} \tag{6.8}$$

with $\mathbf{X} \in \mathbb{K}^{M \times K}$ the output data matrix and the matrices $\mathbf{G}^{(l)} \in \mathbb{K}^{M \times R}$ and $\mathbf{S}^{(l)} \in \mathbb{K}^{K \times R}$ defined element-wise as $g_{mr}^{(l)} = g_{mr}[l]$ and $s_{kr}^{(l)} = s_r[k-l]$ for $0 \leq l \leq L$, respectively. Also, $\mathbf{G} = \begin{bmatrix} \mathbf{G}^{(0)} & \mathbf{G}^{(1)} & \cdots & \mathbf{G}^{(L)} \end{bmatrix} \in \mathbb{K}^{M \times R(L+1)}$ and $\mathbf{S} = \begin{bmatrix} \mathbf{S}^{(0)} & \mathbf{S}^{(1)} & \cdots & \mathbf{S}^{(L)} \end{bmatrix} \in \mathbb{K}^{K \times R(L+1)}$ has a block-Toeplitz structure as illustrated in Figure 6.2. Note that BSI reduces to BSS if $L = 0$. We ignore noise for notational convenience in the derivation of our method. Its influence will be examined in subsection 6.4.2 by means of simulations.

The proposed method reshapes the columns of $\mathbf{X}$, i.e., the observed outputs at time k are put into matrices which are subsequently stacked in a tensor, as shown in Figure 6.2. In general, the columns can be reshaped into Nth-order tensors which are then stacked in a tensor of order $N + 1$. If the system coefficients admit a low-rank model, the BSI problem can be reformulated as a structured flower decomposition of the tensorized observed output data. We will now discuss the different aspects of the method in more detail.

6.3.2 Low-rank coefficient vectors

In large-scale applications vectors and matrices are often compressible, meaning that they can be described in terms of much fewer parameters than the total number of values [190]. Often, the tensor representation of such a vector or matrix allows a low-rank approximation, enabling a possibly very compact model when using higher-order tensors [21], [96], [123]. We denote vectorized low-rank tensors as *low-rank coefficient vectors*. Importantly, the system coefficients in large-scale BSI can often be represented or well approximated by such low-rank tensor models. We show that the exploitation of this low-rank structure in large-scale convolutive BSI enables a unique identification of both the system coefficients and the inputs.

Mathematically, we reshape the rth coefficient vector $\mathbf{g}_r^{(l)}$ in (6.8) into a $(I \times J)$ matrix $\mathbf{G}_r^{(l)}$ such that $\mathrm{vec}(\mathbf{G}_r^{(l)}) = \mathbf{g}_r^{(l)}$ with $M = IJ$. Our working hypothesis states that the matricized coefficient vectors admit a low-rank representation:

$$\mathbf{G}_r^{(l)} = \sum_{p=1}^{P_r^{(l)}} \mathbf{a}_{pr}^{(l)} \otimes \mathbf{b}_{pr}^{(l)} = \mathbf{A}_r^{(l)} \mathbf{B}_r^{(l)^{\mathrm{T}}} \tag{6.9}$$

with $\mathbf{a}_{pr}^{(l)} \in \mathbb{K}^{I}$ and $\mathbf{b}_{pr}^{(l)} \in \mathbb{K}^{J}$. This is equivalent with assuming $\mathbf{g}_r^{(l)}$ can be

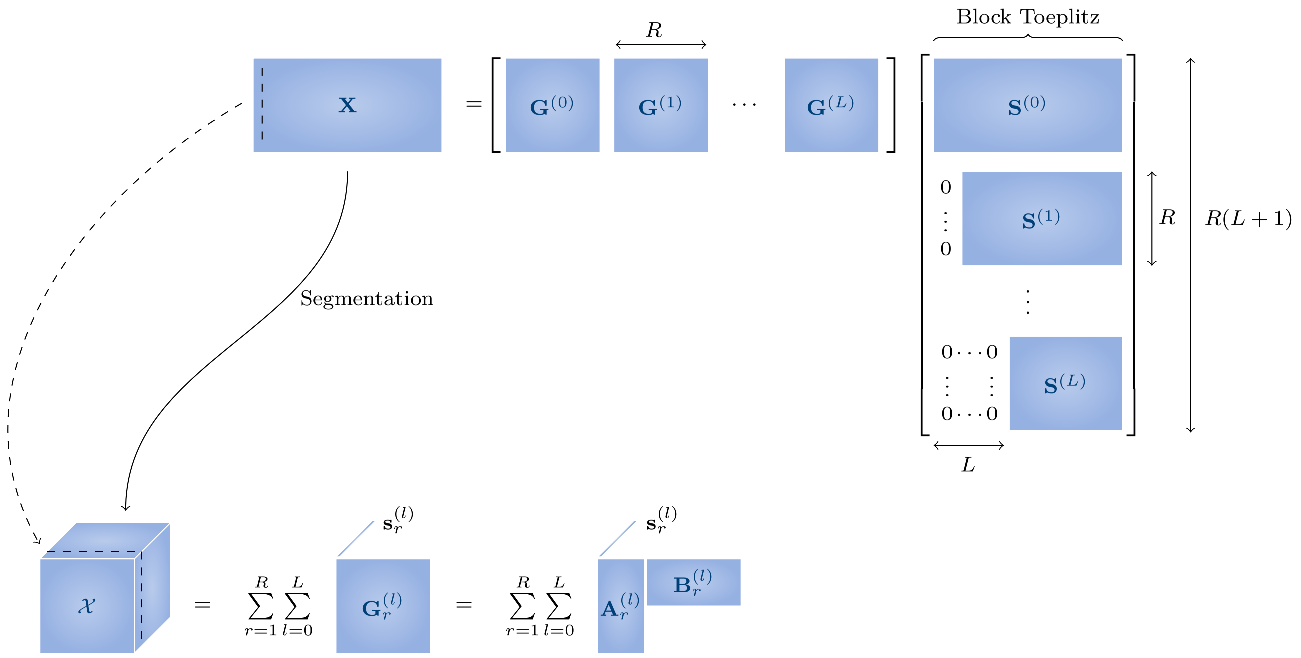


Figure 6.2: Blind System Identification (BSI) can be reduced to the computation of a structured block term decomposition (BTD) by means of segmentation if the coefficients allow a low-rank representation, enabling a unique solution. *Here, each column of the output data matrix* $\mathbf{X}$ *is reshaped into a matrix and then stacked into a third-order tensor* $\mathcal{X}$*. The reshaped system coefficients appear in the first and second mode, while the inputs appear in the third mode. The latter has a block-Toeplitz structure due to the convolutive nature of the system. Hence, this particular tensorization reformulates BSI as a structured tensor decomposition. In this particular example the decomposition is a BTD in multilinear rank-*$(P_r^{(l)}, P_r^{(l)}, 1)$ *terms.*

written as a sum of Kronecker products:

$$\mathbf{g}_r^{(l)} = \operatorname{vec}(\mathbf{G}_r^{(l)}) = \sum_{p=1}^{P_r^{(l)}} \mathbf{b}_{pr}^{(l)} \otimes \mathbf{a}_{pr}^{(l)}.$$

This strategy clearly enables a compact representation of the coefficients as we need only $P_r^{(l)}(I + J - 1)$ parameters. For example, the number of parameters is one order of magnitude lower than the total number of values M when $I \approx J$. Even more compact representations can be obtained by reshaping the coefficients into higher-order tensors, as we will see later.

Exponential polynomials can be used to model a wide variety of signals in many applications. For example, the autonomous behavior of linear systems can be described by (complex) exponentials and, permitting coinciding poles, exponential polynomials. Importantly, the working hypothesis of low-rank coefficient vectors holds exactly for exponential polynomials [21]. We can show this by linking our approach to Hankelization which is a deterministic tensorization technique for BSS [51], [62]. Consider, e.g., an exponential $f(\xi) = z^\xi$ evaluated in $0 \leq \xi \leq 7$. Construct the (4×5) Hankel matrix $\mathbf{H}$ of the resulting vector. Clearly, this matrix has rank one:

$$\mathbf{H} = \begin{bmatrix} 1 & z & z^2 & z^3 & z^4 \\ z & z^2 & z^3 & z^4 & z^5 \\ z^2 & z^3 & z^4 & z^5 & z^6 \\ z^3 & z^4 & z^5 & z^6 & z^7 \end{bmatrix} = \begin{bmatrix} 1 \\ z \\ z^2 \\ z^3 \end{bmatrix} \begin{bmatrix} 1 & z & z^2 & z^3 & z^4 \end{bmatrix}.$$

The (4×2) matrix $\mathbf{G}$, obtained by reshaping the same vector consists of a subset of the columns of the Hankel matrix $\mathbf{H}$:

$$\mathbf{G} = \begin{bmatrix} 1 & z^4 \\ z & z^5 \\ z^2 & z^6 \\ z^3 & z^7 \end{bmatrix} = \begin{bmatrix} 1 \\ z \\ z^2 \\ z^3 \end{bmatrix} \begin{bmatrix} 1 & z^4 \end{bmatrix}.$$

Clearly, $\mathbf{G}$ also has rank one. This idea can be generalized as follows. Consider a vector $\mathbf{f} \in \mathbb{K}^M$ and its matricized version $\mathbf{G} \in \mathbb{K}^{I \times J}$. Consider also the Hankelized version $\mathbf{H} \in \mathbb{K}^{I \times J_h}$ of $\mathbf{f}$ defined element-wise as $h_{ij_h} = f_{i+j_h-1}$ with $M = I + J_h - 1$. Clearly, we have that $\mathbf{G} = \mathbf{HQ}$ with $\mathbf{Q} \in \mathbb{K}^{J_h \times J}$ the selection matrix defined by $\mathbf{q}_j = \mathbf{e}_{(j-1)I+1}$ for $1 \leq j \leq J$, meaning that the columns of $\mathbf{G}$ form a subset of the columns of $\mathbf{H}$. It is well-known that $\mathbf{H}$ has low rank if the underlying functions are exponential polynomials [21], [51]. It is clear that if $\mathbf{H}$ has low rank then $\mathbf{G}$ has low rank as well, while $\mathbf{G}$ offers a more compact representation than $\mathbf{H}$.

General periodic signals can also be reshaped into low-rank matrices. Consider a nonzero signal with period T, i.e., $f(\xi) = f(\xi + T)$. Collect M samples in a vector $\mathbf{f} \in \mathbb{K}^M$ such that $f_\xi = f(\xi)$ for $1 \leq \xi \leq M$. Assume $M = TW$ with W the number of periods. If we reshape $\mathbf{f}$ into a $(T \times W)$ matrix $\mathbf{G}$, then the rank of $\mathbf{G}$ equals one *regardless* of the type of signal (e.g., discontinuities are allowed). Analogously, if we reshape $\mathbf{f}$ into a $(\frac{T}{2} \times 2W)$ matrix, i.e., each column contains one half of a period, then the rank is *at most* two. Hence, the rank is in general at most R if we reshape $\mathbf{f}$ into a $(\frac{T}{R} \times RW)$ matrix, meaning that each column contains $\frac{1}{R}$-th of a period. Conversely, if we obtain a $(RT \times \frac{W}{R})$ matrix, each column contains a multiple of the period, and the rank is one. In practice, however, the period is typically unknown or may have been estimated inaccurately. Hence, it is interesting to investigate how this influences the rank of $\mathbf{G}$. For example, reshape $\mathbf{f}$ into a $((T-1) \times \lfloor \frac{M}{T-1} \rfloor)$ matrix $\mathbf{G}$. In that case, the transpose of $\mathbf{G}$ is a submatrix of the circulant matrix $\mathbf{C}$ constructed from $\mathbf{f}$, denoted as $\mathbf{C} = \text{circ}(\mathbf{f})$, i.e., we have $\mathbf{G}^{\text{T}} = \mathbf{C}_{1:\lfloor \frac{M}{T-1} \rfloor, 1:T-1}$ in MATLAB-like notation. This is illustrated in Figure 6.3. We now use the following property of circulant matrices [217]:

Property 1. Consider a circulant matrix $\tilde{\mathbf{C}} = \text{circ}(\mathbf{c}) \in \mathbb{K}^{T \times T}$ with $\mathbf{c} \in \mathbb{K}^T$ one period of a T-periodic signal $\mathbf{f} \in \mathbb{K}^M$ such that $M = TW$. The matrix $\tilde{\mathbf{C}}$ has full rank if $\mathbf{c}$ contains T nonzero frequency components. The circulant matrix $\mathbf{C} = \text{circ}(\mathbf{f}) \in \mathbb{K}^{M \times M}$ has rank T because $\mathbf{C} = \mathbf{1}_{W \times W} \otimes \tilde{\mathbf{C}}$.

From Property 1 it follows that the rank of $\mathbf{G}$ equals $T-1$. Let us now consider the more general case where the estimate for the period $\hat{T}$ is given by $l(T-k)$.

Theorem 4. Consider the $(I \times J)$ reshaping $\mathbf{G}$ of a T-periodic signal $\mathbf{f} \in \mathbb{K}^M$ such that $M \geq IJ$. Assume one period $\mathbf{c} \in \mathbb{K}^T$ contains T nonzero frequency components. Consider also two integers k and l with $l > 0$. If $I = l(T-k)$ and $J = \lfloor \frac{M}{l(T-k)} \rfloor$, then the rank of $\mathbf{G}$ equals one if $k = 0$. If $k \neq 0$, we have

$$r(\mathbf{G}) \leq \begin{cases} \frac{T}{\gcd(T,kl)} & \text{if} \gcd(T,kl) > 1, \\ \frac{T}{\gcd(T,l)} & \text{if} \gcd(T,l) > 1, \\ \frac{T}{\gcd(T,k)} & \text{if} \gcd(T,k) > 1, \\ \min(T,I,J) & \text{else.} \end{cases}$$

Proof. As mentioned earlier, we have $r(\mathbf{G}) = 1$ if $k = 0$ and $l > 0$. For $k > 0$ it can be verified that $\mathbf{G}^{\text{T}}$ is a submatrix of the circulant matrix $\mathbf{C} = \text{circ}(\mathbf{f}) \in \mathbb{K}^{M \times M}$, i.e., we have that

$$\mathbf{G}^{\text{T}} = \mathbf{C}_{1:kl:kl\left\lfloor \frac{M}{l(T-k)} \right\rfloor, 1:l(T-k)}, \tag{6.10}$$

similar to the example above that was illustrated in Figure 6.3. From Property 1 we know that $r(\mathbf{C}) = T$, i.e., $\mathbf{C}$ contains T linearly independent

$$\mathbf{C} = \begin{bmatrix} \overset{\mathbf{G}^{\mathrm{T}}}{\boxed{\begin{matrix}1&2&3\\4&1&2\\3&4&1\\2&3&4\end{matrix}}} & \begin{matrix}4&1&\cdots&4\\3&4&\cdots&3\\2&3&\cdots&2\\1&2&\cdots&1\end{matrix}\\ \begin{matrix}1&2&3\\ \vdots&\vdots&\vdots\\2&3&4\end{matrix} & \begin{matrix}4&1&\cdots&4\\ \vdots&\vdots&\ddots&\vdots\\1&2&\cdots&1\end{matrix}\end{bmatrix}$$

Figure 6.3: The transpose of a reshaped periodic signal is embedded in its ciruclant matrix. *Consider a reshaping of a T-periodic signal* $\mathbf{f}$ *into a matrix* $\mathbf{G}$ *with dimensions* $(\hat{T} \times \lfloor \frac{M}{\hat{T}} \rfloor)$*. The transpose of* $\mathbf{G}$ *equals a submatrix of the circulant matrix* $\mathbf{C} = \mathit{circ}(\mathbf{f})$*. This is illustrated for a periodic signal* $\mathbf{f} = \begin{bmatrix}1, 2, 3, 4, 1, \cdots, 4\end{bmatrix} \in \mathbb{K}^M$ *with* $T = 4$*,* $W = 3$*, and* $M = TW$ *using* $\hat{T} = T - 1$*.*

rows. However, we only select the first $l(T-k)$ values of each row, meaning that the rows of $\mathbf{G}^{\mathrm{T}}$ are not necessarily linearly independent. First, take $l = 1$. In that case one can see that in (6.10) we select every kth row of $\mathbf{C}$. Hence, if $\gcd(T,k) > 1$, the rank of $\mathbf{G}$ equals *at most* $\frac{T}{\gcd(T,k)}$. If $\gcd(T,k) = 1$, we select $T - k = I$ rows, hence, the rank is bounded by the minimal dimension of $\mathbf{G}$, i.e., $r(\mathbf{G}) \leq \min(I,J)$. Next, take $l > 1$. In that case we select every klth row of $\mathbf{C}$, hence, the rank of $\mathbf{G}$ equals at most $\frac{T}{\gcd(T,kl)}$ if $\gcd(T,kl) > 1$. If $\gcd(T,kl) = 1$, but $\gcd(T,l) > 1$ or $\gcd(T,k) > 1$, then $r(\mathbf{G}) \leq \frac{T}{\gcd(T,l)}$ or $r(\mathbf{G}) \leq \frac{T}{\gcd(T,k)}$, respectively. Finally, if $\gcd(T,kl) = \gcd(T,k) = \gcd(T,l) = 1$, then $r(\mathbf{G}) \leq \min(T,I,J)$ because we select at most T linearly independent rows of $\mathbf{C}$ or the rank is bounded by the dimensions. If $k < 0$, $\mathbf{G}^{\mathrm{T}}$ is a submatrix of the *left* circulant matrix and one can make a similar derivation as above. □

Corollary 1. Consider a T-periodic signal $\mathbf{f} \in \mathbb{K}^M$ that satisfies Property 1. The rank of the $(I \times J)$ reshaped version $\mathbf{G}$ is bounded by $1 \leq r(\mathbf{G}) \leq T$ for any choice of I and J.

Note that Corollary 1 implies that the reshaped version $\mathbf{G}$ of a T-periodic signal is a low-rank matrix if the period T is small compared to the number of samples M.

So far we have discussed signals that exactly admit a low-rank representation. However, low-rank models are powerful models for more general compressible signals as well. This has been thoroughly discussed in [21]. For example, Gaussians, sigmoids, sincs, rational, and hyperbolic functions can typically be well approximated by low-rank models. This is because the singular value spectrum of the matricized version of such functions is often fast decaying, meaning that only few rank-1 terms are needed for a good

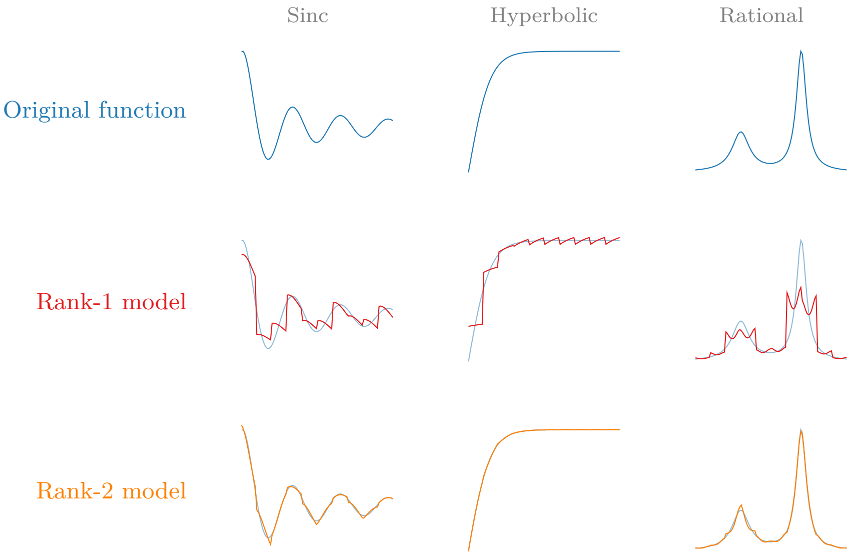


Figure 6.4: A low-rank approximation of a reshaped smooth function often provides an accurate representation. Increasing the rank of the model improves the approximation. *We illustrate for a sinc, a hyperbolic tangent, and a rational function evaluated in 100 equidistant samples in* $\left[0, 1\right]$. *We reshaped the original vectors into* (10×10) *matrices and subsequently approximated them by a low-rank matrix by truncating the singular value decomposition. The reconstructed functions are obtained by vectorizing the resulting rank-*1 *and rank-*2 *matrices. The rank-*2 *model approximately coincides with the original function.*

approximation. This is illustrated in Figure 6.4. Explicit bounds on the approximation error have been reported in [21] for functions that admit a good polynomial approximation.

More generally, one can reshape the coefficient vectors into a higher-order tensor instead of a matrix, allowing an even more compact representation [18], [21]. Indeed, we only need $P_r^{(l)}(\sum_{n=1}^{N} I_n - N + 1)$ parameters instead of $M = \prod_{n=1}^{N} I_n$ to model the (r, l)th coefficient vector. Clearly, the number of parameters decreases logarithmically with the order N of the tensor representation of $\mathbf{g}_r^{(l)}$ and increases proportionally with the number of rank-1 terms $P_r^{(l)}$. Mathematically, we have

$$\mathcal{G}_r^{(l)} = \sum_{p=1}^{P_r^{(l)}} \mathbf{u}_{pr}^{(1,l)} \otimes \mathbf{u}_{pr}^{(2,l)} \otimes \cdots \mathbf{u}_{pr}^{(N,l)} \tag{6.11}$$

with $\mathbf{u}_{pr}^{(n,l)} \in \mathbb{K}^{I_n}$ for $1 \leq n \leq N$. The number of rank-1 terms $P_r^{(l)}$ may be different for different r and for l. Note that this is in fact a PD as in (6.1). Equivalently, the tensorized coefficient vectors can be written as sums of Kronecker products

$$\mathbf{g}_r^{(l)} = \operatorname{vec}(\mathcal{G}_r^{(l)}) = \sum_{p=1}^{P_r^{(l)}} \mathbf{u}_{pr}^{(N,l)} \otimes \mathbf{u}_{pr}^{(N-1,l)} \otimes \cdots \otimes \mathbf{u}_{pr}^{(1,l)}. \tag{6.12}$$

6.3.3 Segmentation and decomposition

We show how BSI can be reformulated as the computation of a structured flower decomposition. The tensor is obtained by using *segmentation* which is a deterministic tensorization technique [18], [21], [62]. The decomposition has a block-Toeplitz structure in the last mode which can be exploited in order to obtain better uniqueness properties and accuracy.

Let us first explain the segmentation approach for the third-order case as depicted in Figure 6.2. We will generalize for order $N > 3$ afterwards. First, we reshape each column of the output data matrix $\mathbf{X}$ in (6.8) into a $(I \times J)$ matrix $\mathbf{X}_k$ such that $\operatorname{vec}(\mathbf{X}_k) = \mathbf{x}_k$ and $M = IJ$. We will discuss the choice of the parameters I and J in more detail in subsection 6.4.4. Next, we stack all the matricized columns in a third-order tensor $\mathcal{X} \in \mathbb{K}^{I \times J \times K}$ such that the kth frontal slice of $\mathcal{X}$ is equal to the kth matricized column of $\mathbf{X}$. This tensorization technique is called segmentation and is a linear operation. This means that the M matricized outputs are linear combinations of the RL shifted sources $\mathbf{s}_r^{(l)}$ using matricized coefficients $\mathbf{G}_r^{(l)} \in \mathbb{K}^{I \times J}$. Hence, it holds that

$$\mathcal{X} = \sum_{r=1}^{R} \sum_{l=0}^{L} \mathbf{G}_r^{(l)} \otimes \mathbf{s}_r^{(l)}.$$

Assume that the system coefficients admit a low-rank model as in (6.9) in order to obtain a BTD in multilinear rank-$(P_r^{(l)}, P_r^{(l)}, 1)$ terms:

$$\mathcal{X} = \sum_{r=1}^{R} \sum_{l=0}^{L} \left(\mathbf{A}_r^{(l)} \mathbf{B}_r^{(l)\mathrm{T}} \right) \otimes \mathbf{s}_r^{(l)}. \tag{6.13}$$

The third factor matrix $\mathbf{S}$ of decomposition (6.13) has a block-Toeplitz structure due to the convolution: $\mathbf{S} = \begin{bmatrix} \mathbf{S}^{(0)} & \mathbf{S}^{(1)} & \cdots & \mathbf{S}^{(L)} \end{bmatrix}$ with $s_{kr}^{(l)} = s_r[k-l]$ for $0 \leq l \leq L$. As such, we have shown that BSI can be solved by means of a structured tensor decomposition. We want to emphasize that it is the working hypothesis of low-rank approximability that has enabled the blind identification. We mentioned uniqueness properties of this particular type of BTD in subsection 6.1.2. We refer the interested reader to [187], [196] for uniqueness properties of block-Toeplitz structured decompositions.

We now generalize the above approach by reshaping the coefficient vectors into tensors instead of matrices, leading to a structured flower decomposition. First, we reshape each column of $\mathbf{X}$ into an Nth-order $(I_1 \times I_2 \times \cdots \times I_N)$ tensor $\mathcal{X}_k$ such that $\operatorname{vec}(\mathcal{X}_k) = \mathbf{x}_k$ and $M = \prod_{n=1}^{N} I_n$. Again we refer to subsection 6.4.4 for a discussion of the choice of I_n. Next, we stack the resulting tensors into a $(N+1)$th-order tensor $\mathcal{X} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N \times K}$ such that the kth tensorized column of $\mathbf{X}$ equals the kth Nth-order "frontal slice" of $\mathcal{X}$, hence,

$$\mathcal{X} = \sum_{r=1}^{R} \sum_{l=0}^{L} \mathcal{G}_r^{(l)} \otimes \mathbf{s}_r^{(l)}.$$

Let us assume that the coefficients admit a low-rank model as in (6.11) in order to obtain the following decomposition:

$$\mathcal{X} = \sum_{r=1}^{R} \sum_{l=0}^{L} \left(\sum_{p=1}^{P_r^{(l)}} \mathbf{u}_{pr}^{(1,l)} \otimes \mathbf{u}_{pr}^{(2,l)} \otimes \cdots \otimes \mathbf{u}_{pr}^{(N,l)} \right) \otimes \mathbf{s}_r^{(l)}. \tag{6.14}$$

Hence, we reformulated BSI as the computation of a block-Toeplitz structured flower decomposition. It is clear that (6.14) reduces to (6.13) if $N = 2$. We discussed uniqueness properties of the flower decomposition in subsection 6.2.2.

The proposed method allows one to uniquely identify both the system coefficients and the inputs of large-scale BSI problems. The compressibility of the coefficients allowed us to rewrite the problem as a tensor decomposition using segmentation. This allows us to benefit from the mild uniqueness properties of tensor decompositions and enables the blind identification. We emphasize that our method is applicable to large-scale FIR systems because of the highly compact representation of the coefficients by means of a higher-order low-rank model. Recall that segmentation is a deterministic tensorization technique, meaning that our method also works for very small sample sizes, see section 6.4.

In contrast to our method, conventional techniques fall short in the large-scale setting. For example, ICA methods that use Qth-order statistics are infeasible when M is large because the number of entries in the resulting tensor is $\mathcal{O}(M^Q)$. Our segmentation-based method reshapes the $(M \times K)$ data matrix into a $(I_1 \times I_2 \times \cdots \times I_N \times K)$ tensor with the same number of entries as in the data matrix. If $I_1 = I_2 = \cdots = I_N = K = I$, the resulting tensor contains $\mathcal{O}(I^{N+1})$ entries, or equivalently $\mathcal{O}(\log_N(M)M)$, which more or less amounts to a decrease of complexity by Q orders of magnitude.

6.3.4 Uniqueness

We derive uniqueness conditions similar to the ones in [51], [65] for the decomposition in (6.8). By ignoring the block-Toeplitz structure on $\mathbf{S}$ in this subsection, we can ignore the superscript l for simplicity and take $1 \leq r \leq R(L+1)$. Assume we have low-rank coefficient vectors of the form:

$$\mathbf{g}_r = \operatorname{vec}(\mathbf{G}_r) = \sum_{p=1}^{P_r} \mathbf{b}_{pr} \otimes \mathbf{a}_{pr}, \tag{6.15}$$

with $\mathbf{a}_{pr} \in \mathbb{K}^I$, $\mathbf{b}_{pr} \in \mathbb{K}^J$, and $R' = \sum_{r=1}^{R} P_r$. Note that $\mathbf{G}_r = \mathbf{A}_r\mathbf{B}_r^{\mathrm{T}}$. We now apply Theorem 3 from subsection 6.1.2.

Theorem 5. Consider a matrix $\mathbf{S} \in \mathbb{K}^{K\times R(L+1)}$ that does not have proportional columns and a matrix $\mathbf{G} \in \mathbb{K}^{M\times R(L+1)}$ of which the columns have structure (6.15). Assume the matrices $\mathbf{A} = \begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2 & \cdots & \mathbf{A}_R\end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix}\mathbf{B}_1 & \mathbf{B}_2 & \cdots & \mathbf{B}_R\end{bmatrix}$ have full column rank. If $M \geq R'^2$ then the decomposition $\mathbf{X} = \mathbf{G}\mathbf{S}^{\mathrm{T}}$ is essentially unique.

Proof. The constraint $M \geq R'^2$ allows us to reshape the columns of $\mathbf{X}$ into $(I \times J)$ matrices $\mathbf{X}_r$ such that $M = IJ$ for $1 \leq r \leq R(L+1)$ with $I, J \geq R'$. The matrices $\mathbf{X}_r$ admit the following decomposition: $\mathbf{X}_r = \mathbf{A}_r\mathbf{B}_r^{\mathrm{T}}$. The matrices $\mathbf{A}_r$ and $\mathbf{B}_r$ have full column rank by definition. The result then follows from Theorem 3. □

We can apply this result to coefficient vectors that can be modeled as sums of exponentials. Element-wise, we have:

$$g_r(\xi) \overset{\text{def}}{=} g_{\xi+1,r} = \sum_{p=1}^{P_r} \alpha_{pr} z_{pr}^{\xi},$$

for $0 \leq \xi \leq M-1$ and $1 \leq r \leq R(L+1)$. One can see that this is a special case of (6.15) as follows. Take $\xi = i + jI$ with $0 \leq i \leq I-1$, $0 \leq j \leq J-1$ and $M = IJ$. Hence, we have

$$g_{\xi+1,r} = \sum_{p=1}^{P_r} \alpha_{pr} z_{pr}^{i+jI} = \sum_{p=1}^{P_r} \alpha_{pr} z_{pr}^{i} z_{pr}^{jI}.$$

By defining $a_{ipr} = z_{pr}^i$ and $b_{jpr} = \alpha_{pr} z_{pr}^{jI}$, we obtain (6.15).

We generalize Theorem 5 for coefficient vectors of the form:

$$\mathbf{g}_r = \operatorname{vec}(\mathcal{G}_r) = \sum_{p}^{P_r} \mathbf{u}_{pr}^{(N)} \otimes \mathbf{u}_{pr}^{(N-1)} \otimes \cdots \otimes \mathbf{u}_{pr}^{(1)} \tag{6.16}$$

with $\mathbf{u}_{pr}^{(n)} \in \mathbb{K}^{I_n}$.

Theorem 6. Consider a matrix $\mathbf{S} \in \mathbb{K}^{K \times R(L+1)}$ that has full column rank and a matrix $\mathbf{G} \in \mathbb{K}^{M \times R(L+1)}$ with structure (6.16). Assume the matrices $\mathbf{U}^{(n)} = \begin{bmatrix} \mathbf{U}_1^{(n)} & \mathbf{U}_2^{(n)} & \cdots & \mathbf{U}_R^{(n)} \end{bmatrix}$, for $1 \leq n \leq N$, have full column rank. If $M \geq R'^2$ then the decomposition $\mathbf{X} = \mathbf{GS}$ is essentially unique.

Proof. Reshape the columns of $\mathbf{X}$ into $(I_1 \times I_2 \times \cdots \times I_N)$ tensors $\mathcal{X}_r$ with $M = \prod_{n=1}^{N} I_n$ for $1 \leq r \leq R(L+1)$. Construct N matrix representations of the form: $\mathbf{X}_r^{(w)} = \mathbf{A}_r^{(w)} \mathbf{B}_r^{(w)\mathrm{T}}$ for $1 \leq w \leq N$ with $\mathbf{A}_r^{(w)} = \bigodot_{\gamma \in \Gamma_w} \mathbf{U}_r^{(\gamma)} \in \mathbb{K}^{I'_w \times P_r}$ and $\mathbf{B}_r^{(w)} = \bigodot_{\upsilon \in \Upsilon_w} \mathbf{U}_r^{(\upsilon)} \in \mathbb{K}^{J'_w \times P_r}$ with $I'_w = \prod_{\gamma \in \Gamma_w} I_\gamma$, and $J'_w = \prod_{\upsilon \in \Upsilon_w} I_\upsilon$. The sets Γ_w and Υ_w satisfy $\Gamma_w \cup \Upsilon_w = \{1, \ldots, N\}$ and $\Gamma_w \cap \Upsilon_w = \emptyset$. The constraint $M \geq R'^2$ allows at least one matrix representation w for which $I'_w, J'_w \geq R'$. The factor matrices $\mathbf{A}_r^{(w)}$ and $\mathbf{B}_r^{(w)}$ have full column rank by definition. The flower decomposition can be interpreted as a coupled BTD in multilinear rank-$(P_r, P_r, 1)$ terms. We know from subsection 6.2.2 that the flower decomposition is unique if one of its BTDs is unique and $\mathbf{S}$ has full column rank. The result then follows from Theorem 3. ☐

Let us now give an example to explain why the uniqueness conditions become milder in the higher-order case. Consider decomposition (6.8) and ignore the block-Toeplitz structure as before. Consider a coefficient matrix $\mathbf{G}$ that has a 6th-order structure ($N = 6$), which we will represent by tensors of increasing order. More specifically, we have a matrix of the form $\mathbf{G} = \mathbf{U}^{(6)} \odot \mathbf{U}^{(5)} \odot \mathbf{U}^{(4)} \odot \mathbf{U}^{(3)} \odot \mathbf{U}^{(2)} \odot \mathbf{U}^{(1)}$ with $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$ and $I_n = 2$, for $1 \leq n \leq N$, using $M = 64$ and $K = 1000$. By applying our segmentation-based approach to $\mathbf{X}$ for increasing $\hat{N}$, we obtain a CPD of an $(\hat{N}+1)$th-order tensor $\mathcal{X}$ of dimensions $(I_1 \times \cdots \times I_{\hat{N}} \times K)$; see Table 6.1 for the values of the dimensions. For $\hat{N} > 2$, one can rework the higher-order CPD into a set of coupled third-order CPDs, similar to the explanation for the flower decomposition in subsection 6.2.2, such that one can use the uniqueness conditions in [184]. In order to illustrate the milder uniqueness conditions for increasing order $\hat{N}$ we check if Corollary 4.13 in [184] is generically satisfied, in the way explained in [196, Section III-B]. The results are shown in Table 6.2. It is clear that the uniqueness conditions are more relaxed for higher $\hat{N}$, i.e., when exploiting more of the intrinsic higher-order structure.

6.3.5 Block-Toeplitz structure

We have shown that convolutive BSI can be reformulated as a block-Toeplitz constrained flower decomposition, assuming low-rank coefficient vectors. Improved uniqueness conditions can be obtained by explicitly exploiting the block-Toeplitz structure of $\mathbf{S}$ in (6.8) as well. Dedicated uniqueness conditions have been presented in [187], [196] for the block-Toeplitz constrained

Table 6.1: Dimensions of the obtained tensor $\mathcal{X}$ by applying segmentation to (6.8) using $M = 64$, $K = 1000$, and a given $\hat{N}$.

$\hat{N}$	$(I_1 \times \cdots \times I_{\hat{N}} \times K)$
2	$8 \times 8 \times 1000$
3	$4 \times 4 \times 4 \times 1000$
4	$2 \times 4 \times 4 \times 2 \times 1000$
5	$2 \times 2 \times 2 \times 2 \times 4 \times 1000$
6	$2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 1000$

Table 6.2: By exploiting more of the intrinsic higher-order structure in problem (6.8), more inputs can be identified. *Here we report the maximum value of R for which Corollary 4.13 in [184] holds for a given $\hat{N}$ and corresponding dimensions given in Table 6.1.*

$\hat{N}$	2	3	4	5	6
R	40	46	49	50	52

(coupled) CPD and the BTD in multilinear rank-$(P_r, P_r, 1)$ terms. In this subsection, we generalize the results for the more general flower decomposition. In other words, we exploit both the higher-order structure *and* the block-Toeplitz structure, enabling more relaxed uniqueness conditions.

Consider the block-Toeplitz decomposition $\mathbf{X} = \mathbf{G}\mathbf{S}^{\mathrm{T}}$ defined in (6.8). We call it essentially unique if any other block-Toeplitz decomposition $\mathbf{X} = \mathbf{M}\mathbf{T}^{\mathrm{T}}$ is related to $\mathbf{X} = \mathbf{G}\mathbf{S}^{\mathrm{T}}$ via a nonsingular matrix $\mathbf{F} \in \mathbb{K}^{R\times R}$ as follows: $\mathbf{G}^{(l)} = \mathbf{M}^{(l)}\mathbf{F}^{\mathrm{T}}$ and $\mathbf{S}^{(l)} = \mathbf{T}^{(l)}\mathbf{F}^{-1}$ for $0 \leq l \leq L$. Several essential uniqueness conditions can be found in [187]. We repeat Lemma 2.4 from [187] as Lemma 2 in this chapter.

Lemma 2. The block-Toeplitz constrained decomposition $\mathbf{X} = \mathbf{G}\mathbf{S}^{\mathrm{T}}$ defined in (6.8) is essentially unique if the matrices $\mathbf{G}$ and $\mathbf{Z} \in \mathbb{K}^{(K-1)\times R(L+2)}$, defined as

$$\mathbf{Z} = \left[\overline{\mathbf{S}}^{(0)}\ \overline{\mathbf{S}}^{(1)}\ \cdots\ \overline{\mathbf{S}}^{(L)}\ \underline{\mathbf{S}}^{(L)}\right]$$

have full column rank. The matrices $\overline{\mathbf{S}}^{(l)}$ and $\underline{\mathbf{S}}^{(l)}$ are equal to $\mathbf{S}^{(l)}$ with the first and last row omitted, respectively.

Remember that the matrix $\mathbf{S}$ in the block-Toeplitz decomposition (6.8) can only be found up to the intrinsic ambiguity $\mathbf{F}$. Hence, we have $\mathbf{S} = \mathbf{T}(\mathbf{I}_{L+1} \otimes \mathbf{F})$ in which $\mathbf{T}$ is a block-Toeplitz matrix with the same column space as $\mathbf{S}$, i.e., $\mathrm{range}(\mathbf{S}) = \mathrm{range}(\mathbf{T})$. As such, we can write (6.8) as

$$\mathbf{X} = \mathbf{G}(\mathbf{I}_{L+1} \otimes \mathbf{F}^{\mathrm{T}})\mathbf{T}^{\mathrm{T}}.$$

A two-step procedure for determining $\mathbf{G}$, $\mathbf{F}$, and $\mathbf{T}$ from $\mathbf{X}$ has been proposed in [187]. First, we determine $\mathbf{T}$ by computing the column space range($\mathbf{X}^{\mathrm{T}}$) = range($\mathbf{S}$), assuming $\mathbf{G}$ and $\mathbf{Z}$ have full column rank, and solving a linear system of equations as explained in [187]. According to Lemma 2, we obtain $\mathbf{S}$ up to the intrinsic block-Toeplitz indeterminacy, i.e., we have $\mathbf{T} = (\mathbf{I}_{L+1} \otimes \mathbf{F}^{-1})\mathbf{S}$. Next, we can determine $\mathbf{G}$ and $\mathbf{F}$ via a coupled decomposition as follows. We have

$$\mathbf{Y} = \mathbf{X}(\mathbf{T}^{\mathrm{T}})^{\dagger} = \mathbf{G}\mathbf{S}^{\mathrm{T}}(\mathbf{T}^{\mathrm{T}})^{\dagger} = \mathbf{G}(\mathbf{I}_{L+1} \otimes \mathbf{F}^{\mathrm{T}}).$$

Let us partition $\mathbf{Y} \in \mathbb{K}^{M \times R(L+1)}$ as $\mathbf{Y} = \begin{bmatrix} \mathbf{Y}^{(0)} & \mathbf{Y}^{(1)} & \cdots & \mathbf{Y}^{(L)} \end{bmatrix}$ in which $\mathbf{Y}^{(l)} \in \mathbb{K}^{M \times R}$. Hence,

$$\mathbf{Y}^{(l)} = \mathbf{G}^{(l)}\mathbf{F}^{\mathrm{T}} \text{ for } 1 \leq l \leq L. \tag{6.17}$$

Equation (6.17) is a coupled decomposition of matrices $\mathbf{Y}^{(l)}$ with a common factor $\mathbf{F}$. One can interpret the block-Toeplitz factorization as a deconvolution, i.e., the convolutive BSI problem has been reduced to an instantaneous BSI problem which takes the form of a coupled decomposition. Decomposition (6.8) can be interpreted as a matrix representation of a block-Toeplitz constrained CPD or BTD in multilinear rank-$(L_r, L_r, 1)$ terms if $\mathbf{G} = \mathbf{B} \odot \mathbf{A}$ or if $\mathbf{G} = \begin{bmatrix} \mathrm{vec}(\mathbf{B}_1\mathbf{A}_1^{\mathrm{T}}) & \cdots & \mathrm{vec}(\mathbf{B}_R\mathbf{A}_R^{\mathrm{T}}) \end{bmatrix}$, respectively.

Here we apply the same idea to the flower decomposition as follows. Consider a coefficient matrix $\mathbf{G}$ with columns defined as in (6.12), resulting in a coupled flower decomposition in (6.17). As explained earlier, a flower decomposition can be written as a coupled BTD in multilinear rank-$(P_r^{(l)}, P_r^{(l)}, 1)$ terms. Hence, each flower decomposition in (6.17) can be written as:

$$\mathbf{Y}^{(w,l)} = \mathbf{G}^{(w,l)}\mathbf{F}^{\mathrm{T}}$$

in which $\mathbf{G}^{(w,l)} \in \mathbb{K}^{I' \times R}$ $(I' = M)$ is defined as

$$\mathbf{G}^{(w,l)} = \begin{bmatrix} \mathrm{vec}(\mathbf{B}_1^{(w,l)}\mathbf{A}_1^{(w,l)^{\mathrm{T}}}) & \cdots & \mathrm{vec}(\mathbf{B}_R^{(w,l)}\mathbf{B}_R^{(w,l)^{\mathrm{T}}}) \end{bmatrix}$$

with $\mathbf{A}_r^{(w,l)} \in \mathbb{K}^{I'_w \times P_r^{(l)}}$, $\mathbf{B}_r^{(w,l)} \in \mathbb{K}^{J'_w \times P_r^{(l)}}$. Or equivalently,

$$\mathbf{Y}^{(w,l)} = \left(\mathbf{A}^{(w,l)} \odot \mathbf{B}^{(w,l)}\right) \mathbf{F}^{(\mathrm{ext,l})^{\mathrm{T}}} \tag{6.18}$$

with $\mathbf{F}^{(\mathrm{ext,l})} = \begin{bmatrix} \mathbf{1}_{P_1^{(l)}}^{\mathrm{T}} \otimes \mathbf{f}_1 \cdots \mathbf{1}_{P_R^{(l)}}^{\mathrm{T}} \otimes \mathbf{f}_R \end{bmatrix}$. Hence, we obtain a coupled BTD and we can use the uniqueness conditions from [184], [187]. First, we define the matrix $\mathbf{V}$:

$$\mathbf{V} = \begin{bmatrix} C_{P+1}(\mathbf{A}^{(1,0)}) \odot C_{P+1}(\mathbf{B}^{(1,0)}) \\ \vdots \\ C_{P+1}(\mathbf{A}^{(1,L)}) \odot C_{P+1}(\mathbf{B}^{(1,L)}) \\ C_{P+1}(\mathbf{A}^{(2,0)}) \odot C_{P+1}(\mathbf{B}^{(2,0)}) \\ \vdots \\ C_{P+1}(\mathbf{A}^{(2,L)}) \odot C_{P+1}(\mathbf{B}^{(2,L)}) \\ \vdots \\ C_{P+1}(\mathbf{A}^{(W,L)}) \odot C_{P+1}(\mathbf{B}^{(W,L)}) \end{bmatrix} \mathbf{P}_{\mathrm{BTD}},$$

in which $\mathbf{A}^{(w,l)} \in \mathbb{K}^{I'_w \times RP}$ and $\mathbf{B}^{(w,l)} \in \mathbb{K}^{J'_w \times RP}$ are defined as

$$\begin{aligned} \mathbf{A}^{(w,l)} &= \begin{bmatrix} \mathbf{A}_1^{(w,l)} & \cdots & \mathbf{A}_R^{(w,l)} \end{bmatrix}, \\ \mathbf{B}^{(w,l)} &= \begin{bmatrix} \mathbf{B}_1^{(w,l)} & \cdots & \mathbf{B}_R^{(w,l)} \end{bmatrix}, \end{aligned}$$

assuming $P_r^{(l)} = P$ for $1 \leq r \leq R$ and $0 \leq l \leq L$.

The matrix $\mathbf{P}_{\mathrm{BTD}}$ is a selection matrix that takes into account that each column of $\mathbf{F}^{(\mathrm{ext})}$ is repeated P times in (6.18), see [186], [187].

Theorem 7. Consider the decomposition of $\mathbf{X}$ in (6.8) in which $\mathbf{G}$ has a structure as in (6.12) with $P_r^{(l)} = P$, for $1 \leq r \leq R$ and $0 \leq l \leq L$, and $\mathbf{S}$ has a block-Toeplitz structure. It is essentially unique if $\mathbf{G}$, $\mathbf{Z}$, $\mathbf{F}$, and $\mathbf{V}$ have full column rank.

Proof. Lemma 2 ensures that we can write the block-Toeplitz decomposition in (6.8) as a coupled flower decomposition. We explained above how this decomposition can be written as a coupled BTD in multilinear rank-$(P_r^{(l)}, P_r^{(l)})$ terms. The results then follows from [187, Theorem II.3]. ☐

By exploiting the block-Toeplitz structure in (6.8), Theorem 7 provides a more relaxed uniqueness condition than Theorem 6. We compare the theorems by checking if the conditions are generically satisfied, in the way explained in [196, Section III-B]. More specifically, we construct random matrices with structure as specified in Theorems 6 and 7. Next, we numerically check for which values of R the conditions hold. The results are shown in Table 6.3 for $M = 1000$, $I_1 = I_2 = I_3 = 10$ $(N = 3)$, and $K = 100$. Clearly, more inputs can be identified by exploiting the available block-Toeplitz structure. The most restrictive constraint in Theorem 7 is the constraint that $\mathbf{Z}$ should have full column rank, hence, the repeated values do not depend on P.

Finally, algebraic methods have been proposed that are guaranteed to find the exact solution in the case of exact decompositions, see [187], [196]. In

Table 6.3: By exploiting the block-Toeplitz structure in (6.8), more relaxed uniqueness conditions can be obtained. *Here, we report the maximum value of R for which Theorems 6 and 7 hold for a given pair (P, L) and $M = 1000$, $I_1 = I_2 = I_3 = 10$, and $K = 100$. Clearly, more inputs can be identified when exploiting the available block-Toeplitz structure.*

L	1			2			3		
P	1	2	3	1	2	3	1	2	3
Theorem 6	15	7	5	10	5	3	7	3	2
Theorem 7	33	33	33	24	24	24	19	19	19

the noisy case, such methods can be used to initialize optimization-based methods.

6.4 Numerical experiments

In subsection 6.4.1 we illustrate our method with a simple proof-of-concept. Next, we inspect the influence of noise and sample size as well as the system order on the accuracy in subsection 6.4.2 and subsection 6.4.3, respectively. In subsection 6.4.4 we discuss parameter selection. In order to compute the relative error on the estimated FIR system coefficients and inputs, they first have to be optimally scaled and permuted with respect to the true ones. This is due to the standard permutation and scaling indeterminacies in BSI. Hence, we define the relative error $\epsilon_{\mathbf{A}}$ as the relative difference in Frobenius norm $||\mathbf{A} - \hat{\mathbf{A}}||_{\mathrm{F}}/||\mathbf{A}||_{\mathrm{F}}$ with $\hat{\mathbf{A}}$ the optimally scaled and permuted estimate for the matrix $\mathbf{A}$. In the experiments we use Gaussian (white) additive noise and i.i.d. zero-mean unit-variance Gaussian random sources of length K unless stated otherwise.

We compute tensor decompositions using least-squares optimization-based methods from Tensorlab [211], [215]. The CPD and the BTD in multilinear rank-$(P_r, P_r, 1)$ terms are computed using `cpd` and `ll1`, respectively, using a GEVD as initialization, see [49], [178], [180], [215]. The (unstructured) decomposition in (rank-$L_r \otimes$ vector) terms is computed with a nonlinear least-squares (NLS) algorithm called `lvec` [18], [21] which is available upon request. We use the GEVD method from subsection 6.2.3 to initialize. For the computation of the block-Toeplitz structured flower decomposition we use a two step procedure. First, we compute the unstructured decomposition as explained earlier. Next, we use this solution to initialize the computation of the block-Toeplitz structured flower decomposition in the SDF framework of Tensorlab [178]. Finally, we mention that for very large tensors one can resort to large-scale algorithms, see [171], [209], [214].

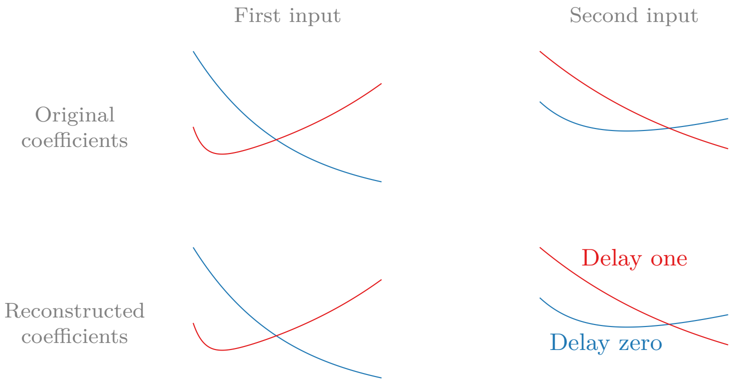


Figure 6.5: By exploiting the low-rank structure of the FIR system coefficients, the coefficient vectors can be perfectly reconstructed (in the noiseless case).

6.4.1 Proof-of-concept

We illustrate the proposed method with a simple proof-of-concept. Consider a large FIR system with $M = 1000$ outputs, $R = 2$ inputs with $K = 100$ samples, and system order $L = 1$. The coefficient vectors $\mathbf{g}_r^{(l)}$ are sums of exponentials: $g_1^{(0)}(\xi) = e^{-2\xi}$, $g_1^{(1)}(\xi) = \frac{1}{4}(5^{-10\xi} + 10^{\frac{\xi}{2}})$, $g_2^{(0)}(\xi) = \frac{1}{3}(e^{\frac{\xi}{2}} + e^{-4\xi})$, and $g_2^{(1)}(\xi) = (\frac{1}{2})^{\frac{3\xi}{2}}$ evaluated in M equidistant samples in $[0, 1]$. We know from subsection 6.3.2 that a sum of P exponentials leads to a reshaped tensor of rank-P. Hence, we use an Nth-order rank-1 approximation for $g_1^{(0)}$ and $g_2^{(1)}$ ($P_1^{(0)} = P_2^{(1)} = 1$) and we use an Nth-order rank-2 approximation for $g_1^{(1)}$ and $g_2^{(0)}$ ($P_1^{(1)} = P_2^{(0)} = 2$). We choose $N = 3$ with $I_n = 10$ for $1 \leq n \leq N$. As such, we decompose the $(10 \times 10 \times 10 \times 100)$ segmented tensor obtained from the observed data matrix $\mathbf{X}$ into a sum of (rank-$P_r^{(l)}$ $\otimes$ vector) terms. We use the two step procedure explained above to compute a solution. Note that we need only $P_r^{(l)}(I_1 + I_2 + I_3 - 2)$ values to model the (r, l)th coefficient vector. More specifically, we need only 28 or 56 values instead of 1000 for a rank-1 or -2 approximation, respectively. Hence, we have compression rates of $1 - P_r^{(l)} \frac{I_1 + I_2 + I_3 - 2}{M} = 97.20\%$ and 94.40%, respectively. Higher compression rates can be attained by further increasing N. Both the original and perfectly reconstructed coefficient vectors are shown in Figure 6.5.

6.4.2 Influence of noise and sample size

Let us illustrate the influence of the noise and the sample size K for the proposed method. Consider a large FIR system with $M = 1000$ outputs,

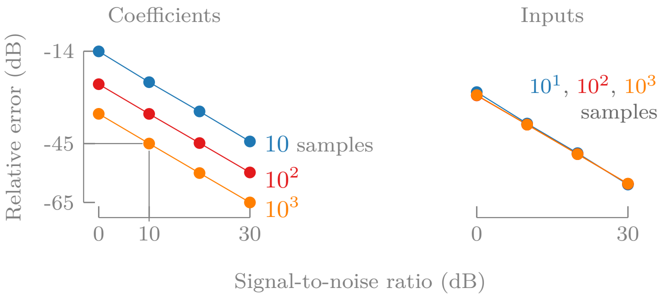


Figure 6.6: The proposed method clearly obtains accurate results in comparison with the SNR for both the coefficients and the inputs, e.g., a relative error of -45 dB for 10 dB SNR. Increasing the number of samples improves the accuracy on the coefficients. This is not the case for the inputs due to the fact that one also has to estimate longer input signals.

$R = 3$ inputs of sample size K, and system order $L = 1$. We vary the SNR from 0 dB to 30 dB in steps of 10 dB and choose $K = 10^i$ for $1 \leq i \leq 3$. The low-rank coefficient vectors are constructed as vectorized low-rank tensors using (6.12). Specifically, we have $N = 2$, $P_r^{(l)} = 2$ for all delays of the first input, $P_r^{(l)} = 1$ for all delays of the other two inputs, and $I = J = 50$ for $1 \leq n \leq 2$ using random zero-mean unit-variance Gaussian-distributed factor vector entries. In Figure 6.6, we report the median across 50 experiments of the relative error on the system coefficients $\epsilon_{\mathbf{G}}$ and the inputs $\epsilon_{\mathbf{S}}$. The results clearly show that the accuracy is very high in comparison with the signal-to-noise ratio (SNR) for both the coefficients and the inputs. Moreover, even a small sample size K leads to accurate results. Also, increasing the sample size K is beneficial for $\epsilon_{\mathbf{G}}$ but has no effect on $\epsilon_{\mathbf{S}}$. This is due to the fact that one also has to estimate longer input signals, which was also observed for instantaneous BSS [21].

6.4.3 Influence of the system order L

We analyze the effect of under- or overestimating the system order L for our segmentation-based method. Consider a large FIR system with $M = 1000$ outputs, $R = 2$ inputs of sample size $K = 100$, and exact system order $L = 2$. The low-rank coefficient vectors are constructed as vectorized rank-1 tensors using (6.12) with $N = 3$, $P_r^{(l)} = 1$ for all r and l, $I_n = 10$ for $1 \leq n \leq 3$, and random zero-mean unit-variance Gaussian-distributed factor vector entries. The SNR is varied from 0 dB to 30 dB in steps of 10 dB. We apply our method for $0 \leq \hat{L} \leq 4$. The relative error on the system coefficients $\epsilon_{\mathbf{G}}$ is defined as $\epsilon_{\mathbf{G}} = ||\hat{\mathbf{G}} - \mathbf{GPD}||_{\text{F}}/||\hat{\mathbf{G}}||_{\text{F}}$ with $\hat{\mathbf{G}} \in \mathbb{K}^{M \times R(\hat{L}+1)}$ and $\mathbf{G} \in \mathbb{K}^{M \times R(L+1)}$. $\mathbf{P} \in \mathbb{K}^{R(L+1) \times R(\hat{L}+1)}$ is the optimal column selection and permutation matrix, and $\mathbf{D} \in \mathbb{K}^{R(\hat{L}+1) \times R(\hat{L}+1)}$ is the optimal scaling matrix. The relative error on the

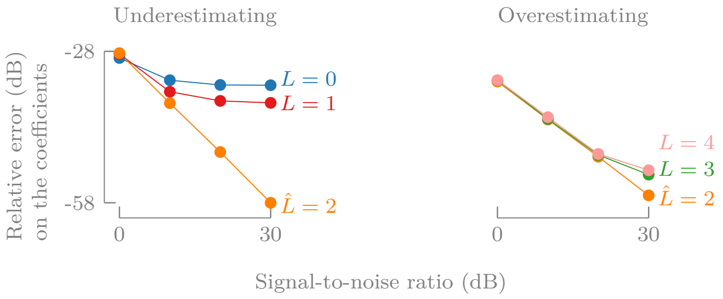


Figure 6.7: Overestimating the exact system order $L = 2$ is not so critical for the accuracy on the coefficients. Underestimating the system order reduces the accuracy but the results are still quite good.

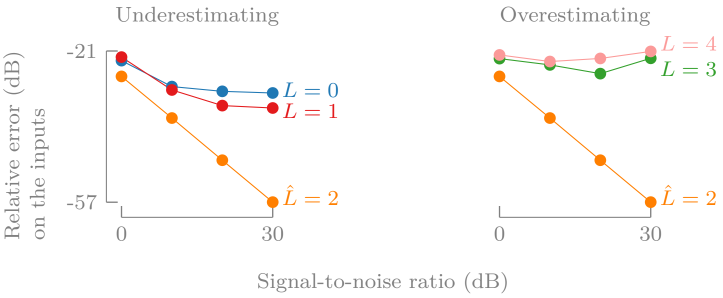


Figure 6.8: Although under- and overestimating the exact system order $L = 2$ decreases the accuracy on the inputs, the former provides slightly more accurate results than the latter.

inputs $\epsilon_{\mathbf{S}}$ is defined in a similar way. We report $\epsilon_{\mathbf{G}}$ and $\epsilon_{\mathbf{S}}$ in Figure 6.7 and Figure 6.8, respectively. While overestimating the system order L is not so critical for $\epsilon_{\mathbf{G}}$, underestimating leads to less accurate results. Both under- and overestimating the system order decreases the accuracy on the inputs, but underestimating leads to slightly more accurate results than overestimating.

In practice, however, one can find a reasonable estimate for the system order L as follows. The multilinear rank of the tensor $\mathcal{X}$ in (6.13) is bounded by $(\sum_{r=1}^{R}\sum_{l=0}^{L} P_r^{(l)}, \sum_{r=1}^{R}\sum_{l=0}^{L} P_r^{(l)}, RL)$. Hence, one can find an estimate for RL, in which R is equal to the number of inputs, by computing the MLSVD of $\mathcal{X}$ and checking the number of significant mode-3 singular values.

6.4.4 Parameter selection

We discuss the choice of the dimensions I and J of the segmentation matrix and the rank of the model $P_r^{(l)}$ for the (r,l)th coefficient vector. What is

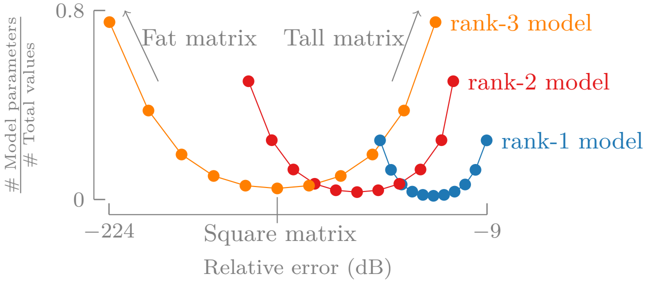


Figure 6.9: Our segmentation-based approach allows a trade-off between compactness and accuracy of the model through the choice of the dimensions and the rank of the model. *What is considered a 'good' choice of parameters depends on the needs in a particular application. If compactness is the objective, a square segmentation matrix is preferred. If accuracy is the objective, a fat matrix clearly outperforms a tall one and better results can be attained by increasing the rank. An explanation for the former phenomenon is illustrated in Figure 6.10 using a rank-1 approximation of two different segmented matrices.*

considered a "good" choice of parameters depends of course on the needs in a particular application. In our case, we are most interested in the compactness and the accuracy of the model. Given those objectives, we discuss a simple example to illustrate good choices and how to obtain them.

Consider a Gaussian with mean 0.5 and standard deviation 0.15 that is uniformly discretized in $M = 2^{14}$ samples in $[0, 1]$. We reshape the resulting vector into an $(I \times J)$ matrix with $I = 2^q$ and $J = 2^{14-q}$ for $2 \leq q \leq 12$ such that $IJ = M$. Subsequently, we compute the best rank-P approximation by truncating the singular value decomposition (SVD) for $P = \{1, 2, 3\}$. Define the normalized number of parameters as the ratio between the number of parameters needed in the model and the total number of values in the original vector, i.e., $\hat{M} = P(I + J - 1)/M$. In Figure 6.9, we report the normalized number of parameters $\hat{M}$ versus the relative error ϵ for a rank-P model.

There is a trade-off between compactness and accuracy:

- The accuracy can be improved by choosing I and J such that $I < J$ rather than $I > J$. In other words, a fat segmentation matrix is better than a tall one for a fixed rank; this is illustrated in Figure 6.10. Hence, segmentation is not symmetric in the modes that it creates.

- Increasing the rank P of the model improves the accuracy, especially when $I < J$.

- A compact model, on the other hand, can be obtained by reshaping into a (nearly) square matrix ($I \approx J$) and choosing P not too large.

In practice, one can first overestimate P and use the above guidelines to find some reasonable segmentation dimensions. Most often the value of P is not

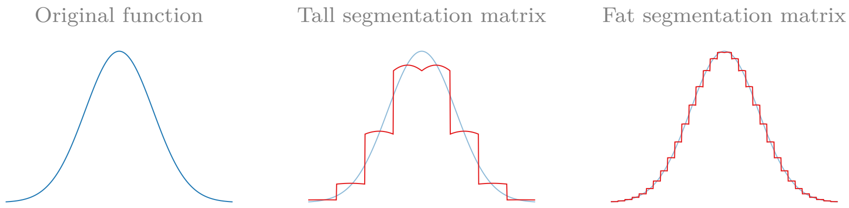


Figure 6.10: By using a fat segmentation matrix, one can often attain a more accurate approximation than by using a tall segmentation matrix. *A tall segmentation matrix often leads to a poor approximation because the original function is divided in only a few segments. In the case of a rank-1 approximation ($P = 1$), each segment is approximated by the same 'long' vector multiplied by a different coefficient. Conversely, the original function is divided in many small segments when using a fat segmentation matrix, leading to an overall good approximation.*

very critical, cf. also [21]. Next, one can repeat the analysis with smaller values of P and further refine the choice of the parameters.

Let us illustrate that overestimation of P is not so critical. Consider a FIR system with $M = 100$ outputs, $R = 2$ inputs with $K = 10$ samples, and system order $L = 1$. The coefficient vectors $\mathbf{g}_r^{(l)}$ are exponentials: $g_1^{(0)}(\xi) = e^{-2\xi}$, $g_1^{(1)}(\xi) = e^{\frac{\xi}{2}}$, $g_2^{(0)}(\xi) = -e^{\xi}$, and $g_2^{(1)}(\xi) = \frac{1}{2}e^{-\xi}$ evaluated in M equidistant samples in $[0, 1]$. It is known that an exponential can be exactly represented by a rank-1 model [21], [51]. However, we overestimate the rank value of the coefficient vectors for the zeroth delay (of both inputs) by one, i.e., we use $P_1^{(0)} = P_2^{(0)} = 2$ and $P_1^{(1)} = P_2^{(1)} = 1$. We take $N = 2$ with $I = J = 10$. The overestimation of the rank value is clearly not so critical: we can perfectly reconstruct the coefficient vectors as shown in Figure 6.11. We also show the spectrum of the low-rank models $\mathbf{G}_1^{(0)}$ and $\mathbf{G}_2^{(0)}$ for coefficient vectors $\mathbf{g}_1^{(0)}$ and $\mathbf{g}_2^{(0)}$, respectively, in Figure 6.12. It is clear that the rank has been overestimated; a rank-1 model would have sufficed.

6.5 Applications

6.5.1 Direction-of-arrival estimation

A uniform rectangular array (URA) is an antenna array with $M = M_x M_y$ antennas that are uniformly spaced in a rectangular grid as depicted in Figure 6.13. There are M_x and M_y antennas in the x- and y-direction, respectively. Let us assume that the output of the mth antenna satisfies (6.7) and that the R inputs impinging on the URA are narrow-band signals. In that case it can be shown that the system coefficients satisfy $\mathbf{g}_r^{(l)} = \mathbf{g}_{x,r}^{(l)} \otimes \mathbf{g}_{y,r}^{(l)}$ with $\mathbf{g}_{x,r}^{(l)}$ and $\mathbf{g}_{y,r}^{(l)}$ defined element-wise as $g_{x,mr}^{(l)} = (\theta_r^{(l)})^{m-1}$

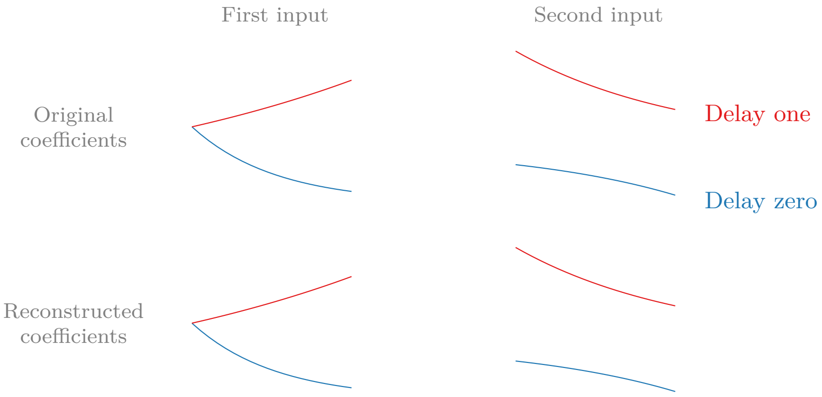


Figure 6.11: Although we overestimate the rank value of the zeroth coefficient vector of both inputs, the FIR system coefficients are perfectly reconstructed in the noiseless case.

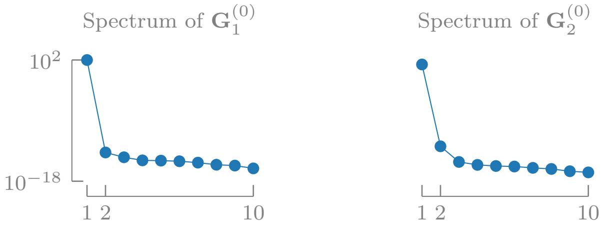


Figure 6.12: The spectra of the obtained low-rank models $\mathbf{G}_1^{(0)}$ and $\mathbf{G}_2^{(0)}$ for the reshaped coefficient vectors of the zeroth delay clearly show a rank-1 model would have sufficed.

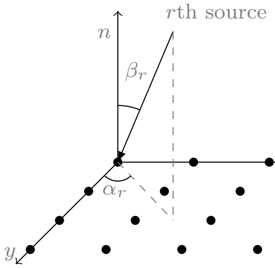


Figure 6.13: Illustration of a uniform rectangular array (URA) with $M = M_x M_y$ antennas: $M_x = 4$ and $M_y = 4$ in the x- and y-direction, respectively. The rth source is impinging on the URA from the far field and is characterized by two angles: the azimuth α_r and elevation β_r relative to the x-axis and the normal, respectively, hence, we have $-90° \leq \alpha_r, \beta_r \leq 90°$.

and $g_{y,mr}^{(l)} = (\phi_r^{(l)})^{m-1}$, respectively. We have $\theta_r^{(l)} = e^{2\pi i \Delta_x \cos(\alpha_r^{(l)}) \sin(\beta_r^{(l)}) \lambda^{-1}}$ and $\phi_r^{(l)} = e^{2\pi i \Delta_y \sin(\alpha_r^{(l)}) \sin(\beta_r^{(l)}) \lambda^{-1}}$ with inter-element spacings denoted by Δ_x and Δ_y, and λ denotes the wavelength. The angle α_r to the x-direction is the azimuth and the angle β_r to the normal is the elevation. Clearly, the system coefficients are Kronecker products of Vandermonde vectors that allow a rank-1 representation. We compare our segmentation-based method with the well-known MUSIC method for DOA estimation [127].

Consider a large square URA with $M = 625$ antennas ($M_x = M_y = 25$) with $R = 2$ inputs, system order $L = 1$, and $K = 100$ samples. Assume Δ_x and Δ_y are both equal to half the wavelength λ. The azimuth and elevation pairs are given by $(\alpha_1^{(0)}, \beta_1^{(0)}) = (-51°, 80°)$, $(\alpha_1^{(1)}, \beta_1^{(1)}) = (55°, -60°)$, $(\alpha_2^{(0)}, \beta_2^{(0)}) = (25°, -20°)$, and $(\alpha_2^{(1)}, \beta_2^{(1)}) = (80°, 51°)$. Recall that the coefficients $\mathbf{g}_r^{(l)}$ are Kronecker products of Vandermonde vectors (which admit a rank-1 representation). Hence, we first reshape each coefficient vector $\mathbf{g}_r^{(l)}$ into a (25×25) matrix $\mathbf{G}_r^{(l)}$ such that $\mathrm{vec}(\mathbf{G}_r^{(l)}) = \mathbf{g}_r^{(l)}$ and so we have $\mathbf{G}_r^{(l)} = \mathbf{g}_{y,r}^{(l)} \otimes \mathbf{g}_{x,r}^{(l)}$. Next, we use a (5×5) second-order rank-1 model for $\mathbf{g}_{x,r}^{(l)}$ and $\mathbf{g}_{y,r}^{(l)}$. This simply leads to an overall fourth-order rank-1 model ($N = 4$) for each $\mathbf{g}_r^{(l)}$ with $I_n = 4$ for $1 \leq n \leq 4$. As such, we only need $\sum_{n=1}^{4} I_n - 3 = 17$ values instead of 625 which means a compression rate of $1 - \frac{\sum_{n=1}^{N} I_n - 3}{M} = 97.28\%$.

We report the median across 100 experiments of the relative errors on the azimuth and elevation angles, denoted as ϵ_α and ϵ_β, respectively, for varying SNR in Figure 6.14. Clearly, segmentation yields more accurate results than MUSIC; note the high accuracy compared to the SNR. The accuracy of MUSIC, however, is bounded by the number of points used to evaluate the 2D MUSIC spectrum [127]. Remember that MUSIC first computes an eigenvalue decomposition of the $(M \times M)$ covariance matrix in order to eval-

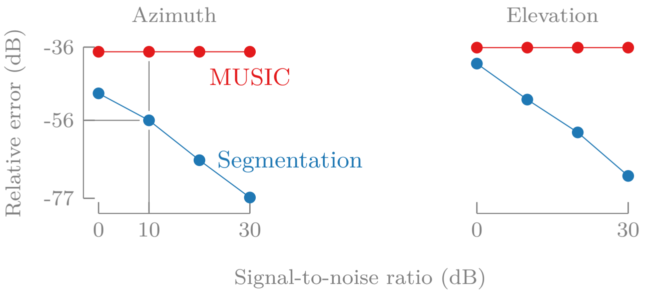


Figure 6.14: Segmentation is clearly more accurate than 2D-MUSIC. *For example, we have a relative error of -56 dB and -36 dB for 10 dB SNR for segmentation and 2D-MUSIC, respectively. Also, the accuracy of segmentation is high compared to the SNR, even at low SNR. The accuracy of 2D-MUSIC, on the other hand, is bounded by the number of points in which the MUSIC spectrum is evaluated. This number has been limited due to the large number of antennas in the large-scale URA under consideration. However, the median computation cost of 2D-MUSIC is still quite high compared to segmentation: 10.04 seconds versus 0.77 seconds on a standard laptop.*

uate the spectrum. The peaks of the spectrum correspond to azimuth and elevation pairs. In order to attain accurate estimates, one has to evaluate the spectrum in many angles which can become computationally expensive. Here, we used 100 equidistant angles in $[-\frac{\pi}{2}, \frac{\pi}{2}]$ for both DOAs to evaluate the spectrum. We limited the number of evaluation points because of the relatively large number of antennas. It is the high computational load of MUSIC for large M that makes it inaccurate or computationally infeasible in a large-scale setting.

It is not unlikely that in a large-scale array a number of antennas will malfunction. This would result in an observed data matrix with a few or even all entries missing in several rows, leading to an incomplete tensor after segmentation. Tensorlab's built-in support for incomplete tensors, however, allows us to compute a decomposition and retrieve the DOAs [215]. Furthermore, the arrays are typically non-uniform in large-scale applications. One way to tackle this problem is to fit an imaginary uniform grid on top of the existing antennas. The resulting URA is very dense and mostly filled with non-existing antennas, resulting in a large and incomplete observed data matrix. However, there exist algorithms with first and second-order convergence that have a computational complexity that is only linear in the number *known* entries [214]. Hence, it is still possible to estimate the DOAs in this case. In Figure 6.15 we report the results for the same experiment as above but now with a (random) fraction of the antennas turned off. Clearly, the estimates are still very accurate, even for low SNR and 90% of the array inactive.

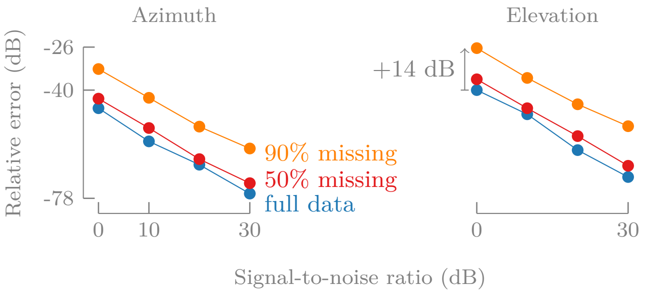


Figure 6.15: The loss in accuracy in a large-scale uniform rectangular array with missing antennas is limited, even for many inactive antennas *and* low SNR. *For example, we have only 14 dB loss in accuracy for the elevation in 0 dB SNR when up to 90% of the antennas are inactive. As such, a segmentation-based approach using incomplete tensor decompositions enables DOA estimation in large-scale grids with a few broken antennas or even non-uniform grids. In the latter case, the grid is "completed" with a dense uniform one that has many missing antennas.*

6.5.2 Neural spike sorting

Spike-sorting refers to the separation of spike trains fired by different neurons from high-density micro-electrode array (HD-MEA) recordings. Often an instantaneous BSS model is assumed and one can use, e.g., independent component analysis (ICA) to extract the spike trains [131]. A convolutive model as in (6.7), however, is typically more accurate because the signals do not propagate instantaneously [117]. Moreover, the assumption of independence is not satisfied when spikes coincide. Also, our method works if only a few samples are available. The amplitude of a spike train typically decreases with a $1/d$ characteristic in which d equals the distance between the neuron and the array. As such, the system coefficients can be assumed low-rank in the high-density setting. We illustrate our method for the separation of simulated coinciding spike trains from a large-scale convolutive mixture.

Consider an array with $M = 1000$ sensors, system order $L = 1$, and $R = 2$ neural spike trains with $K = 270$ samples. Assume the system coefficients can be modeled using the $1/d$ characteristic as mentioned above, i.e., we have:

$$g_r^{(l)}(\xi) = \frac{a_r^{(l)}}{\sqrt{\alpha_r^2 + \left(\frac{\xi-\beta_r}{b_r^{(l)}}\right)^2}}$$

evaluated in M equidistant samples in $[0, 1]$. α_r equals the distance between the array and the rth neuron. $(x - \beta_r)$ equals the distance between an electrode of the array and the electrode with maximum amplitude for the rth neuron. $a_r^{(l)}$ and $b_r^{(l)}$ are shape coefficients for the rth neuron and lth delay. We use $\alpha_1 = 0.1$, $\alpha_2 = 0.05$, $\beta_1 = 0.2$, and $\beta_2 = 0.7$. We use the

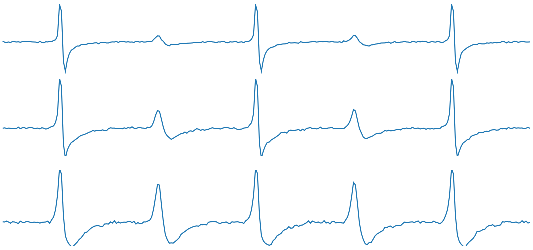


Figure 6.16: Simulated outputs of a high-density microelectrode array for measuring neuronal activity using simulated spike trains as inputs.

following shape coefficient pairs $(a_1^{(0)}, b_1^{(0)}) = (1,1)$, $(a_1^{(1)}, b_1^{(1)}) = (0.5, 0.1)$, $(a_2^{(0)}, b_2^{(0)}) = (1,1)$, and $(a_2^{(1)}, b_2^{(1)}) = (0.7, 0.05)$. The inputs are spike trains of length K with the spikes modeled as a linear combination of two rational functions:

$$s_1(t) = \frac{0.7}{\frac{(t-0.5)^2}{0.01^2}+1} - \frac{0.3}{\frac{(t-0.54)^2}{0.03^2}+1},$$

$$s_2(t) = \frac{0.3}{\frac{(t-0.5)^2}{0.045^2}+1} - \frac{0.15}{\frac{(t-0.6)^2}{0.13^2}+1}.$$

The SNR is 30 dB. Some outputs are shown in Figure 6.16. A rank-2 approximation is sufficient to accurately model the smooth system coefficients, i.e., we have $P_r^{(l)} = 2$ for $1 \leq r \leq 2$ and $0 \leq l \leq 1$. We choose $I = 20$ and $J = 50$ so that we have a fat reshaping, see subsection 6.4.4. In Figure 6.17 we see an excellent separation of the spike trains, even for coinciding spikes and small sample size. Although we do not exploit the periodicity of the inputs, it is possible to use a two-fold segmentation approach consisting of segmentation steps along both the input and mixing level as in [21].

6.6 Conclusion

In this chapter, we presented the first BSI method that is applicable to large-scale FIR systems. The key idea is that in large-scale applications the system coefficients are often compressible because there is a lot of structure that can be exploited. We used low-rank tensor models to approximate the tensorized system coefficients in a compact way, enabling large-scale BSI. We showed that our method reduces BSI to a structured decomposition of a tensor obtained by applying segmentation on the measured outputs. This enabled a unique identification of the system and reconstruction of the inputs; no additional assumptions, such as independence, are needed on the inputs. The method even works well when only a few samples are available because it

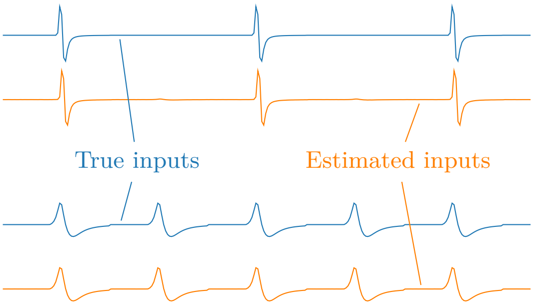


Figure 6.17: Our segmentation-based approach obtains an excellent separation of a convolutive mixture of simulated spike trains stemming from high-density microelectrode arrays for measuring neuronal activity.

is deterministic. The decomposition that we used is a generalization of a particular block term decomposition called the flower decomposition. We discussed uniqueness properties and proposed a new algebraic method to compute it. We also discussed uniqueness properties when incorporating the block-Toeplitz structure of the decomposition. Our method proved viable for DOA estimation in large-scale URAs with possibly broken antennas and even in non-uniform arrays. Also, we demonstrated the use of our method for convolutive spike sorting problems.

7

Large-scale autoregressive system identification using Kronecker product equations

ABSTRACT | By exploiting the intrinsic structure and/or sparsity of the system coefficients in large-scale system identification, one can enable efficient processing. In this chapter, we employ this strategy for large-scale single-input multiple-output autoregressive system identification by assuming the coefficients can be well approximated by Kronecker products of smaller vectors. We show that the identification problem can then be reformulated as the computation of a Kronecker product equation, allowing one to use optimization-based and algebraic solvers.

This chapter is a slightly adapted version of M. Boussé and L. De Lathauwer, "Large-scale autoregressive system identification using Kronecker product equations", in *2018 6th IEEE Global Conference on Signal and Information Processing (GlobalSIP 2018, Anaheim, California, USA)*, Nov. 2018, pp. 1348–1352.

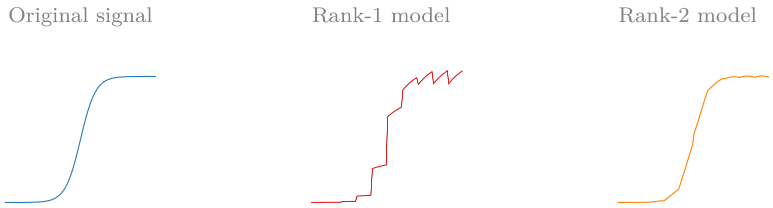


Figure 7.1: Low-rank matrix or tensor models can often provide a parsimonious and accurate representation for smooth data.

7.1 Introduction

System identification is an important engineering problem in various applications, allowing us to model various systems using input *and* output data. [137]. Recently, there is a growing interest in *large-scale* system identification because of an increasing number and density of antennas or sensors in fields such as array processing, telecommunications, and (biomedical) signal processing [13], [107], [130]. In order to tackle such large-scale problems, the intrinsic structure and/or sparsity of the data can be exploited by means of parsimonious models.

Large-scale data is often *compressible*, or, in other words, it can often be described in terms of much fewer parameters than the total number of values [190]. Well-known examples are (exponential) polynomials, rational functions, and smooth and periodic functions [20], [21], [51], [65], [95], [96], [123]. Explicitly exploiting the intrinsic compactness of this type of data, enables efficient processing in large-scale applications. Popular compact models are low-rank matrix and tensor decompositions; see [41], [125], [168] and references therein. A well-known approach consists of reshaping a large-scale vector or matrix into a tensor which can then be modeled using a low-rank approximation [99]; this is illustrated for a sigmoid[1] in Figure 7.1. This approach has successfully allowed one to handle various large-scale applications in tensor-based scientific computing and (blind) system identification [20], [21], [96], [156], [159], [173], [174].

We adopt a similar strategy for autoregressive (AR) system identification [137], [139], enabling large-scale applications. In this chapter, we limit ourselves to single-input multiple-output (SIMO) AR models with Kronecker product constrained coefficients. Although this particular structure corre-

---

[1] We evaluated a sigmoid of the form $f(\xi) = 1/(1+e^{-20(\xi-1/2)})$ in 100 equidistant samples in $[0,1]$ and then reshaped the vector of length 100 containing the values into a (10×10) matrix. We computed a low-rank model by truncating the singular value decomposition and the reconstructions are obtained by vectorizing the resulting rank-1 and rank-2 matrices.

sponds to a rank-1 model, as we will explain later, it can already provide an accurate and compact model while allowing us to explain the basic principles within the space restrictions of this chapter. More specifically, we show that by explicitly exploiting the Kronecker structure, AR system identification can be reformulated as a type of *Kronecker product equation* (KPE). By the latter we mean a linear system of equations with a Kronecker product constrained solution, which has already been applied successfully in various applications [23], [25], [26]. A *generic* framework for this type of problems was developed in [26], allowing us to use optimization-based and algebraic solvers and formulate generic uniqueness conditions.

In the remainder of this section we give an overview of the notation that is used in this chapter, several basic definitions, and KPEs. We derive our method for large-scale SIMO AR system identification using KPEs in section 7.2. In section 7.3, we analyze our method via several numerical experiments. We conclude the chapter and discuss future work in section 7.4.

7.1.1 Notations and basic definitions

Vectors, matrices, and tensors are denoted by bold lowercase, bold uppercase, and calligraphic letters, respectively. The vectorization of an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ ($\mathbb{K}$ meaning $\mathbb{R}$ or $\mathbb{C}$), denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \cdots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1+\sum_{k=1}^{N}(i_k-1)J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$. The inverse operation of $\operatorname{vec}(\cdot)$ is $\operatorname{unvec}(\cdot)$. We indicate the nth element in a sequence by a superscript between parentheses, e.g., $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$. The outer and Kronecker product are denoted by $\otimes$ and $\otimes$, respectively. They are related through a vectorization: $\operatorname{vec}(\mathbf{a} \otimes \mathbf{b}) = \mathbf{b} \otimes \mathbf{a}$.

The rank of a tensor is equal to the minimal number of rank-1 tensors that generate the tensor as their sum. A rank-1 tensor is defined as the outer product of non-zero vectors.

7.1.2 Kronecker product equation

A KPE is a linear system of equations with a Kronecker product constrained solution that has been applied successfully in various domains [23], [25], [26]. In this chapter, we limit ourselves to problems with the following Kronecker product structure:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{with} \quad \mathbf{x} = \mathbf{v} \otimes \mathbf{u}, \tag{7.1}$$

in which $\mathbf{A} \in \mathbb{K}^{K \times Q}$, $\mathbf{x} \in \mathbb{K}^{Q}$, and $\mathbf{b} \in \mathbb{K}^{K}$. The solution $\mathbf{x}$ can be expressed as a Kronecker product $\mathbf{v} \otimes \mathbf{u}$ with $\mathbf{u} \in \mathbb{K}^{I}$ and $\mathbf{v} \in \mathbb{K}^{J}$ such that $Q = IJ$. More generally, $\mathbf{x}$ can be constrained by a Kronecker product of N non-zero vectors:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{with} \quad \mathbf{x} = \bigotimes_{n=1}^{N} \mathbf{u}^{(n)},$$

in which $\mathbf{u}^{(n)} \in \mathbb{K}^{I_n}$ and $Q = I_1 I_2 \cdots I_N$. Importantly, a KPE is a special case of a linear system of equations with a tensor-decomposition constrained solution [26]. This type of problems could be solved by first solving the system without structure and subsequently computing a rank-1 model of the tensorized version of the solution. This approach works well if $\mathbf{A}$ has full column rank, but, in contrast to the methods in [26], fails when $\mathbf{A}$ is rank deficient or when $K < Q$, i.e., in the underdetermined case. The methods in [26] compute the least-squares (LS) solution of (7.1).

7.2 Large-scale SIMO autoregressive system identification using KPEs

By exploiting the intrinsic structure or sparsity of a model, one can enable large-scale system identification. Here, we show that large-scale SIMO AR system identification can be reformulated as a particular type of KPE by exploiting the hypothesized Kronecker product structure of the coefficients. First, we define AR system identification and Kronecker constrained coefficients in subsection 7.2.1 and subsection 7.2.2, respectively. Next, we derive our KPE-based method for large-scale SIMO AR system identification in subsection 7.2.3.

7.2.1 Autoregressive system identification

Consider a MIMO AR model with Q outputs, P exogenous inputs, and system order L, that relates the outputs $y_q[k]$ using the following discrete difference equation:

$$\sum_{l=0}^{L} \sum_{q=1}^{Q} g_{pq}[l]\ y_q[k-l] = x_p[k] + n_p[k] \quad \text{for} \quad 1 \leq k \leq K \tag{7.2}$$

The AR coefficients are given by $g_{pq}[l]$ for $0 \leq l \leq L$, the pth exogenous input is denoted by $x_p[k]$ and the additive *white* noise is given by $n_p[k]$. Assuming we have $K + L$ samples, the model in (7.2) can be expressed in matrix form as follows:

$$\sum_{l=0}^{L} \mathbf{G}^{(l)} \mathbf{Y}^{(l)} = \mathbf{X} + \mathbf{N} \tag{7.3}$$

with $\mathbf{G}^{(l)}$ the lth $(P \times Q)$ coefficient matrix and $\mathbf{Y}^{(l)}$ the lth $(Q \times K)$ output matrix, which are defined element-wise as $g_{pq}^{(l)} = g_{pq}[l]$ and $y_{qk}^{(l)} = y_q[k-l]$, respectively, for $0 \leq l \leq L$. The input and noise matrix $\mathbf{X}$ and $\mathbf{N}$ both have dimensions $(P \times K)$. Note that one typically assumes $P = Q$ when considering the MIMO case; see, e.g., [137], [139] and references therein. The formulation in (7.3), however, is more general because we allow that $P \neq Q$. In this chapter, we limit ourselves to *single-input multiple-output* systems, i.e., we have $P = 1$ and $Q > 1$. In that case, the AR model with exogenous inputs (ARX) in (7.3) reduces to:

$$\sum_{l=0}^{L} \mathbf{g}^{(l)^{\mathrm{T}}} \mathbf{Y}^{(l)} = \mathbf{x}^{\mathrm{T}} + \mathbf{n}^{\mathrm{T}}$$

with coefficients $\mathbf{g}^{(l)} \in \mathbb{K}^{Q}$ and input and noise $\mathbf{x}, \mathbf{n} \in \mathbb{K}^{K}$. The noise is omitted in the derivation of our method for notational convenience, but its influence is examined in section 7.3.

7.2.2 Kronecker constrained system coefficients

Large-scale data can often be compactly modeled because of some intrinsic structure or sparsity of the data. In this chapter, we take a similar approach as in [20], [21]: we assume the (large-scale) AR coefficients admit, or, can be well approximated by, a Kronecker product of N non-zero vectors, enabling a possibly very compact representation for large N. Consider the following Kronecker product structure for $\mathbf{g}^{(l)} \in \mathbb{K}^{Q}$:

$$\mathbf{g}^{(l)} = \mathbf{b}^{(l)} \otimes \mathbf{a}^{(l)}, \quad \text{for} \quad 0 \leq l \leq L, \tag{7.4}$$

with non-zero vectors $\mathbf{a}^{(l)} \in \mathbb{K}^{I^{(l)}}$ and $\mathbf{b}^{(l)} \in \mathbb{K}^{J^{(l)}}$ such that $Q = I^{(l)}J^{(l)}$, for $0 \leq l \leq L$. Clearly, this approach allows for a compact representation of the coefficients: we need only $(I^{(l)} + J^{(l)} - 1)$ values instead of $Q = I^{(l)}J^{(l)}$ to represent $\mathbf{g}^{(l)}$. Interestingly, constraint (7.4) corresponds to a rank-1 assumption on a matricized version of $\mathbf{g}^{(l)}$, i.e., we have: $\operatorname{mat}\left(\mathbf{g}^{(l)}\right) = \mathbf{a}^{(l)^{\mathrm{T}}} \mathbf{b}^{(l)} = \mathbf{a}^{(l)} \circledcirc \mathbf{b}^{(l)}$. More generally, one can consider a Kronecker product of N non-zero vectors:

$$\mathbf{g}^{(l)} = \bigotimes_{n=1}^{N} \mathbf{u}^{(n,l)}, \quad \text{for} \quad 0 \leq l \leq L, \tag{7.5}$$

with $\mathbf{u}^{(n,l)} \in \mathbb{K}^{I_n^{(l)}}$ such that $Q = \prod_{n=1}^{N} I_n^{(l)}$, for $0 \leq l \leq L$. Increasing N, enables even more compact representations because we need only $\sum_{n=1}^{N} I_n^{(l)} - N + 1$ values instead of $Q = \prod_{n=1}^{N} I_n^{(l)}$ to represent $\mathbf{g}^{(l)}$. For example, if $I_n^{(l)} = I$ for $1 \leq n \leq N$ and $0 \leq l \leq L$, the number of unknown variables reduces from $\mathcal{O}(LI^N)$ to $\mathcal{O}(LNI)$. For $N > 2$, constraint (7.5) corresponds to a rank-1 assumption on a tensorized version of $\mathbf{g}^{(l)}$, i.e., we have: $\operatorname{unvec}\left(\mathbf{g}^{(l)}\right) = \mathbf{u}^{(1,l)} \circledcirc \mathbf{u}^{(2,l)} \circledcirc \cdots \circledcirc \mathbf{u}^{(N,l)}$. A detailed analysis on how to choose the dimensions of the vectors in the Kronecker product can be found in [20], [21].

7.2.3 Large-scale AR system identification as a KPE

By explicitly exploiting the Kronecker structure in the model, one can reformulate AR system identification as the computation of a (structured) KPE, allowing one to use optimization-based and algebraic solvers and formulate (generic) uniqueness conditions; see [26]. We illustrate this as follows.

Assuming the AR coefficients $\mathbf{g}^{(l)}$, for $0 \leq l \leq L$, can be modeled by a simple Kronecker product as in (7.4), we obtain:

$$\sum_{l=0}^{L} \left(\mathbf{b}^{(l)} \otimes \mathbf{a}^{(l)}\right)^{\mathrm{T}} \mathbf{Y}^{(l)} = \mathbf{x}^{\mathrm{T}}. \tag{7.6}$$

By taking the transpose, one can see that (7.6) reduces to:

$$\sum_{l=0}^{L} \mathbf{Y}^{(l)^{\mathrm{T}}} \left(\mathbf{b}^{(l)} \otimes \mathbf{a}^{(l)}\right) = \mathbf{x}. \tag{7.7}$$

For $L = 0$, this model reduces to a KPE of the form (7.1). For $L > 0$, the model in (7.7) is a straightforward generalization where the right-hand side equals a sum of $L + 1$ matrix-times-Kronecker-product terms. More generally, one can consider a Kronecker product of N non-zero vectors as in (7.5), obtaining:

$$\sum_{l=0}^{L} \mathbf{Y}^{(l)^{\mathrm{T}}} \left(\bigotimes_{n=1}^{N} \mathbf{u}^{(n,l)}\right) = \mathbf{x},$$

which enables higher compression rates, as explained before. As such, large-scale AR system identification is reformulated as the computation of a particular KPE. Additionally, the matrix $\tilde{\mathbf{Y}} = \left[\mathbf{Y}^{(0)^{\mathrm{T}}}, \mathbf{Y}^{(1)^{\mathrm{T}}}, \cdots, \mathbf{Y}^{(L)^{\mathrm{T}}}\right] \in \mathbb{K}^{K \times (L+1)Q}$ has a block-Toeplitz structure due to the convolutive nature of the ARX model. This structure can be exploited to speed up KPE algorithms and relax uniqueness conditions; see [26].

7.3 Experiments

We illustrate our method for various scenarios: 1) a proof-of-concept experiment in which we use exponentials as coefficient vectors, 2) an analysis of the influence of noise and sample size on the accuracy, and 3) an analysis of under- or overestimating the system order. For each experiment, we simulate a random SIMO ARX system by fixing both the coefficients and the outputs, and then constructing an input that satisfies the model in (7.3). We use i.i.d. zero-mean unit-variance Gaussian random outputs for each experiment and we further specify the particular coefficient definition in each experiment de-

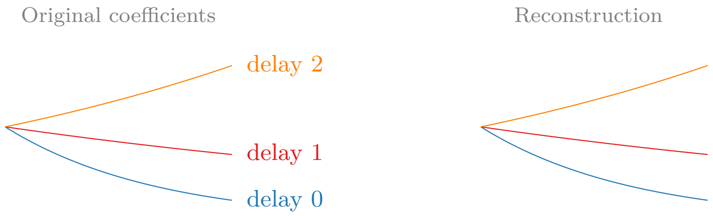


Figure 7.2: By exploiting the rank-1 structure, the autoregressive coefficients are perfectly reconstructed (in the noiseless case).

scription. When considering the noisy case, we use Gaussian (white) additive noise. We define the relative error $\epsilon_{\mathbf{A}}$ as the relative difference in Frobenius norm $\|\mathbf{A} - \hat{\mathbf{A}}\|_{\mathrm{F}}/\|\mathbf{A}\|_{\mathrm{F}}$. We use an adapted version of the non-linear LS algorithms with random initialization from [26] in order to solve KPEs.

7.3.1 Proof-of-concept experiment

Perfect reconstruction of the AR coefficients can be obtained in the noiseless case by exploiting the intrinsic rank-1 structure. We illustrate this for a large-scale SIMO ARX system of order $L = 2$ with $Q = 2500$ outputs and sample size $K = 600$. The coefficients $\mathbf{g}^{(l)}$ are defined as exponentials of length Q; more specifically, we have $g^{(0)}(\xi) = \frac{1}{10}\exp^{-3\xi/2}$, $g^{(1)}(\xi) = \frac{1}{10}(0.5)^{\xi/2}$, and $g^{(2)}(\xi) = \frac{1}{10}\exp^{(\xi/2)}$ uniformly sampled in $[0, 1]$. It is well-known that exponentials can be exactly represented by a rank-1 structure [20], [21], [51], validating the model in (7.7). We choose $N = 2$ and $I_1^{(l)} = I_2^{(l)} = I = 50$, for $0 \leq l \leq L$. Hence, we need only $(2I - 1) = 99$ values to model an AR coefficient vector instead of 2500, which amounts to a compression rate of $1 - \frac{I_1+I_2-1}{P} = 96.04\%$. The original coefficients and their reconstruction, up to machine precision, are shown in Figure 7.2.

7.3.2 Influence of noise and sample size on the accuracy

While increasing the sample size K improves the accuracy of the estimates, even a small number of samples can lead to accurate results. Also, the accuracy is quite high in comparison to the signal-to-noise ratio (SNR). We illustrate this for a large-scale SIMO ARX system of order $L = 5$ with $Q = 500$ outputs. We construct the rank-1 coefficient vectors as vectorized third-order rank-1 tensors using i.i.d. zero-mean unit-variance Gaussian random factor vectors. We use the following dimensions for the coefficient vectors: $(I_1^{(0)}, I_2^{(0)}, I_3^{(0)}) = (I_1^{(1)}, I_2^{(1)}, I_3^{(1)}) = (20, 5, 5)$, $(I_1^{(2)}, I_2^{(2)}, I_3^{(2)}) = (I_1^{(3)}, I_2^{(3)}, I_3^{(3)}) = (25, 5, 4)$, $(I_1^{(4)}, I_2^{(4)}, I_3^{(4)}) = (I_1^{(5)}, I_2^{(5)}, I_3^{(5)}) = (50, 5, 2)$. We choose $K = 1210$ and 12100, which is equal to five and fifty times the

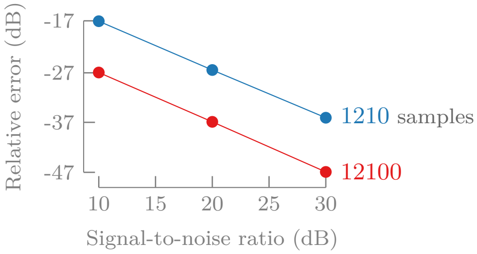


Figure 7.3: While our method obtains accurate results with respect to the signal-to-noise ratio, increasing the number of samples further improves the accuracy of the coefficient estimates.

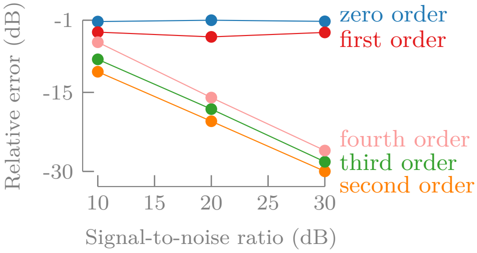


Figure 7.4: While overestimation slightly reduces the accuracy, underestimating the system order fails to give accurate results.

number of unknown coefficients (242 values), and use an SNR equal to 10, 20, or 30 dB. The median results across fifty random experiments are illustrated in Figure 7.3.

7.3.3 Influence of the system order on the accuracy

Although the accuracy of the estimates slightly reduces, overestimating the system order L is not so critical. However, underestimating the order fails to give accurate results. We illustrate this for a large-scale SIMO ARX system of order $L = 2$ with $Q = 100$ outputs. We construct the rank-1 coefficient vectors as vectorized rank-1 matrices using i.i.d. zero-mean unit-variance Gaussian random factor vectors. We use the following dimensions for the coefficient vectors: $I_1^{(l)} = 20$ and $I_2^{(l)} = 5$, for $0 \leq l \leq L$. We choose $K = 50$, which is equal to twice the number of unknown coefficients. We use an SNR of 10, 20, or 30 dB. While estimating the coefficients, we vary the system order between zero and four. The median results across fifty random experiments are shown in Figure 7.4.

7.4 Conclusion and future work

We have presented a method for AR system identification that enables large-scale applications by explicitly exploiting the hypothesized structure/sparsity of the system coefficients. In this chapter, we have shown that the identification problem can be reformulated as the computation of a KPE, allowing one to use optimization-based solvers. Numerical experiments have shown that our method performs well in noisy conditions and that over-estimation of the system order is not so critical.

In follow-up work, we will address 1) the multiple-input multiple-output case, 2) the explicit exploitation of the block-Toeplitz structure in the computation and the uniqueness conditions, and 3) sum-of-Kronecker-products constrained coefficients. The latter means that we approximate a matricized or tensorized version of the coefficient vectors by a low-rank model instead of rank-1 model as explained in this chapter.

Face recognition as a Kronecker product equation

8

ABSTRACT | Various parameters influence face recognition such as expression, pose, and illumination. In contrast to matrices, tensors can be used to naturally accommodate for the different modes of variation. The multilinear singular value decomposition (MLSVD) then allows one to describe each mode with a factor matrix and the interaction between the modes with a coefficient tensor. In this chapter, we show that each image in the tensor satisfying an MLSVD model can be expressed as a structured linear system called a Kronecker product equation (KPE). By solving a similar KPE for a new image, we can extract a feature vector that allows us to recognize the person with high performance. Additionally, more robust results can be obtained by using multiple images of the same person under different conditions, leading to a coupled KPE. Finally, our method can be used to update the database with an unknown person using only a few images instead of an image for each combination of conditions. We illustrate our method for the extended Yale Face Database B, achieving better performance than conventional methods such as Eigenfaces and other tensor-based techniques.

This chapter is a slightly adapted version of M. Boussé, N. Vervliet, O. Debals, and L. De Lathauwer, "Face recognition as a Kronecker product equation", in *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2017, Curaçao, Dutch Antilles)*, Dec. 2017, pp. 276–280. The associated paper was nominated for a best student paper award at CAMSAP 2017. The figures have been updated for consistency.

8.1 Introduction

Face recognition is an important problem in computer vision with many applications within domains such as information security, surveillance, and biometric identification [220]. Although many recognition systems use matrix-based methods, face recognition is inherently a multidimensional problem due to variations in facial expression, pose, illumination conditions, etc. [203]. Linear algebra is of limited use because it only captures a single variation using a mode of a matrix. For example, the well-known Eigenfaces method stacks vectorized images in the second mode, obtaining a matrix with modes pixels × persons [194]. Although some methods have tried to accommodate for different conditions [37], [89], the multidimensional structure remains a challenging problem for matrix-based methods.

Recently, tensor tools have gained increasing popularity in signal processing and machine learning applications [41], [168]. Tensors are higher-order generalizations of vectors (first order) and matrices (second order). The higher-order structure allows one to explicitly accommodate for the multidimensional structure of facial images: each mode of a tensor can represent a single variation of the image [203]. For example, a set of (vectorized) images of several persons under different illumination conditions can be represented by a third-order tensor with modes pixels × illuminations × persons. An important tensor tool is the multilinear singular value decomposition (MLSVD) of a higher-order tensor which is a generalization of the well-known singular value decomposition (SVD) [54]. The MLSVD allows one to approximately represent the tensor by a set of factor matrices that are each related to a single mode and a core tensor that explains the interaction between the different modes. This type of representation is also used in TensorFaces, enabling improved accuracy in face recognition in comparison with conventional techniques such as Eigenfaces [204]. Several other tensor-based methods have been proposed; see [102], [104].

In this chapter, we explain that tensor-based face recognition using the MLSVD model can be expressed as a Kronecker product equation (KPE). A KPE is a linear system of equations with a Kronecker product constrained solution for which the authors have developed a generic framework in [26]. We show that by solving a KPE for a new unlabeled image, one can obtain a feature vector that enables better recognition rates than conventional methods. In practice, the robustness can be improved by coupling multiple images of the same person under different conditions, leading to a set of KPEs that are coupled. Additionally, our method allows one to add a new unknown person using only a few images instead of an image for each combination of conditions. We illustrate our method for the extended Yale Face Database B which contains facial images of multiple persons under different illuminations [88]. Our KPE-based method achieves higher performance than conventional Eigenfaces and the tensor-based approach in [204].

We conclude this section with an overview of the notation and basic definitions. We also define the MLSVD and KPEs. Next, we reformulate face recognition as a KPE in section 8.2. We apply our approach to a real-life dataset in section 8.3.

8.1.1 Notations and basic definitions

We denote vectors, matrices, and tensors by bold lowercase, e.g., $\mathbf{a}$, bold uppercase, e.g., $\mathbf{A}$, and calligraphic letters, e.g., $\mathcal{A}$, respectively. A natural extension of the rows and columns of a matrix, is a mode-n vector of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, defined by fixing every index except the nth, e.g., $\mathbf{a}_{i_1 \cdots i_{n-1} : i_{n+1} \cdots i_N}$. A mode-$n$ unfolding of $\mathcal{A}$ is a matrix $\mathbf{A}_{(n)}$ with the mode-n vectors as its columns (following the ordering convention in [125]). The vectorization of $\mathcal{A}$, denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \cdots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1 + \sum_{k=1}^{N} (i_k - 1) J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$. We indicate the nth element in a sequence by a superscript between parentheses, e.g., $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$.

The outer and Kronecker product are denoted by $\otimes$ and $\otimes$, respectively. The mode-n product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $\mathbf{B} \in \mathbb{R}^{J_n \times I_n}$ is a tensor $\mathcal{A} \cdot_n \mathbf{B} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots I_N}$ and is defined element-wise as $(\mathcal{A} \cdot_n \mathbf{B})_{i_1 \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} a_{i_1 i_2 \cdots i_N} b_{j_n i_n}$. Hence, each mode-$n$ vector of the tensor $\mathcal{A}$ is multiplied with the matrix $\mathbf{B}$, i.e., $(\mathcal{A} \cdot_n \mathbf{B})_{(n)} = \mathbf{B}\mathbf{A}_{(n)}$.

8.1.2 Multilinear singular value decomposition

The MLSVD of a higher-order tensor is a generalization of the SVD of a matrix [41], [54], [168]. The MLSVD writes a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ as the product

$$\mathcal{A} = \mathcal{S} \cdot_1 \mathbf{U}^{(1)} \cdot_2 \mathbf{U}^{(2)} \cdots \cdot_n \mathbf{U}^{(N)},$$

in which $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ is a unitary matrix, $n = 1, \ldots, N$, and the core tensor $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is ordered and all-orthogonal; see [54] for more details. The mode-n rank of an Nth-order tensor is equal to the rank of the mode-n unfolding. The multilinear rank of the tensor is equal to the tuple of mode-n rank values. The MLSVD is related to the low-multilinear rank approximation (LMLRA) and the Tucker decomposition (TD); see, e.g., [54], [125], [214]. The decomposition has been used successfully in applications such as compression and dimensionality reduction [59], [125].

8.1.3 (Coupled) Kronecker product equations

A KPE is a linear system of equations with a Kronecker product constrained solution. Here, we limit ourselves to problems with the following simple

Kronecker product structure:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{with} \quad \mathbf{x} = \mathbf{v} \otimes \mathbf{u}, \tag{8.1}$$

in which $\mathbf{A} \in \mathbb{R}^{M \times K}$, $\mathbf{x} \in \mathbb{R}^{K}$, and $\mathbf{b} \in \mathbb{R}^{M}$. The solution $\mathbf{x}$ can be expressed as a Kronecker product $\mathbf{v} \otimes \mathbf{u}$ with $\mathbf{u} \in \mathbb{R}^{I}$ and $\mathbf{v} \in \mathbb{R}^{J}$ such that $K = IJ$. As a matter of fact, a KPE is a simple case of a linear system with a solution that can be represented as a matrix or tensor decomposition [26]. Expression (8.1) can be solved by first solving the system without structure and subsequently decomposing a matricized version of the solution. This approach works well if $\mathbf{A}$ has full column rank, but, in contrast to the methods in [26], fails when $\mathbf{A}$ is rank deficient or in the underdetermined case. The methods in [26] compute the least-squares (LS) solution of (8.1).

A coupled KPE (cKPE) is a set of KPEs that have a common coefficient vector. We limit ourselves to cKPEs of the form:

$$\mathbf{A}(\mathbf{v} \otimes \mathbf{u}^{(q)}) = \mathbf{b}^{(q)} \quad \text{for} \quad q = 1, \dots, Q,$$

with $\mathbf{A} \in \mathbb{R}^{M \times K}$, $\mathbf{v} \in \mathbb{R}^{I}$, $\mathbf{u}^{(q)} \in \mathbb{R}^{J}$, and $\mathbf{b}^{(q)} \in \mathbb{R}^{M}$ such that $K = IJ$. By defining $\mathbf{U} \in \mathbb{R}^{J \times Q}$ with $\mathbf{u}_q = \mathbf{u}^{(q)}$ and $\mathbf{B} \in \mathbb{R}^{M \times Q}$ with $\mathbf{b}_q = \mathbf{b}^{(q)}$, we obtain

$$\mathbf{A}(\mathbf{v} \otimes \mathbf{U}) = \mathbf{B}.$$

8.2 Face recognition using KPEs

8.2.1 Tensorization and MLSVD model

Higher-order tensors can explicitly accommodate for the multidimensional nature of facial images by representing each variation by a mode of the tensor [203], [204]. Although our method can be used for any combination of variations, we illustrate the strategy for the following particular case. Consider a set of facial images of J persons taken under I different illumination conditions. Each image is represented by a matrix of size $M_x \times M_y$ with M_x and M_y pixels in the x- and y-direction, respectively. All *vectorized* images of length $M = M_x M_y$ are stacked into a third-order tensor $\mathcal{D} \in \mathbb{R}^{M \times I \times J}$ with modes pixels (px) × illuminations (i) × persons (p).

Next, we compute a truncated MLSVD of the tensor $\mathcal{D}$:

$$\mathcal{D} \approx \mathcal{S} \cdot_1 \mathbf{U}_{\text{px}} \cdot_2 \mathbf{U}_{\text{i}} \cdot_3 \mathbf{U}_{\text{p}}, \tag{8.2}$$

in which $\mathbf{U}_{\text{px}} \in \mathbb{R}^{M \times P}$, $\mathbf{U}_{\text{i}} \in \mathbb{R}^{I \times R}$, and $\mathbf{U}_{\text{p}} \in \mathbb{R}^{J \times L}$ form an orthonormal basis for the pixel, illumination, and person mode, respectively. The interaction between the different modes is expressed by the core tensor $\mathcal{S} \in \mathbb{R}^{P \times R \times L}$.

Each row of $\mathbf{U}_\mathrm{p}$, denoted by $\mathbf{c}_p^\mathrm{T}$, can be interpreted as the coefficients for person p and each row of $\mathbf{U}_\mathrm{i}$, denoted by $\mathbf{c}_i^\mathrm{T}$, can be interpreted as the coefficients for illumination i.

8.2.2 Kronecker product equation

Each mode-1 fiber of $\mathcal{D}$ in (8.2) corresponds to an image and can be modeled by a KPE as follows. Consider a vectorized image $\mathbf{d} \in \mathbb{R}^M$ for a *particular* person p and illumination i:

$$\mathbf{d} = (\mathcal{S} \cdot_1 \mathbf{U}_\mathrm{px}) \cdot_2 \mathbf{c}_i^\mathrm{T} \cdot_3 \mathbf{c}_p^\mathrm{T}, \tag{8.3}$$

$$\mathbf{d} = \mathbf{U}_\mathrm{px}\mathbf{S}_{(1)} \left(\mathbf{c}_p \otimes \mathbf{c}_i\right). \tag{8.4}$$

Expression (8.4) is the mode-1 unfolding of (8.3) and is a KPE: each $\mathbf{d}$ is a linear combination of the columns of $\mathbf{U}_\mathrm{px}\mathbf{S}_{(1)}$ with Kronecker product constrained coefficients $(\mathbf{c}_p \otimes \mathbf{c}_i)$.

Consider a set of facial images of the same person under Q different illuminations, leading to a set of coupled KPEs that share the coefficient vector in the *person mode*:

$$\mathbf{d}^{(q)} = \mathbf{U}_\mathrm{px}\mathbf{S}_{(1)}(\mathbf{c}_p \otimes \mathbf{c}_i^{(q)}) \quad \text{for} \quad q = 1, \ldots, Q. \tag{8.5}$$

By stacking all vectorized images $\mathbf{d}^{(q)}$ and illumination coefficient vectors $\mathbf{c}_i^{(q)}$ into a matrix $\mathbf{D} \in \mathbb{R}^{M\times Q}$ and $\mathbf{C}_i \in \mathbb{R}^{R\times Q}$, respectively, we obtain

$$\mathbf{D} = \mathbf{U}_\mathrm{px}\mathbf{S}_{(1)}(\mathbf{c}_p \otimes \mathbf{C}_i).$$

8.2.3 Face recognition

We explain how to recognize a person in a (set of) facial image(s) under a new illumination condition using (c)KPEs. First, we construct a tensor $\mathcal{D}$ by stacking a set of facial images of different persons under different illuminations in the way explained in subsection 8.2.1. Second, we compute the truncated MLSVD of $\mathcal{D}$, obtaining factor matrices $\mathbf{U}_\mathrm{px}$, $\mathbf{U}_\mathrm{i}$, and $\mathbf{U}_\mathrm{p}$, and core tensor $\mathcal{S}$. Every (vectorized) image of $\mathcal{D}$ can then be expressed as a KPE as explained in subsection 8.2.2. Next, consider a new, unlabeled facial image $\mathbf{d}^{(\mathrm{new})}$ of a known person. In order to recognize the person in the image, we solve the following KPE using the algorithms from [26]:

$$\mathbf{d}^{(\mathrm{new})} = \mathbf{U}_\mathrm{px}\mathbf{S}_{(1)} \left(\mathbf{c}_p^{(\mathrm{new})} \otimes \mathbf{c}_i^{(\mathrm{new})}\right), \tag{8.6}$$

obtaining estimates $\tilde{\mathbf{c}}_p^{(\mathrm{new})}$ and $\tilde{\mathbf{c}}_i^{(\mathrm{new})}$ for the coefficient vectors. We compare $\tilde{\mathbf{c}}_p^{(\mathrm{new})}$ with the rows of $\mathbf{U}_\mathrm{p}$ using the Frobenius norm of the difference (after fixing scaling and sign invariance). One can then recognize the person in the image by assigning the label corresponding to the closest match. In other words, the estimated coefficient vector for the person mode $\tilde{\mathbf{c}}_p^{(\text{new})}$ acts as a feature vector and $\mathbf{U}_\text{p}$ acts as a database. More robust results can be obtained by using images under multiple illumination conditions and coupling the KPEs as in (8.5).

In contrast to our method, the tensor-based approach in [204] solves (8.6) by fixing the illumination coefficients to a particular illumination. More specifically, the approach solves (8.6) by taking $\mathbf{c}_i^{(\text{new})}$ equal to a row of $\mathbf{U}_\text{i}$, reducing (8.6) to a linear system of equations for *each* illumination condition. Every estimate is then compared with $\mathbf{U}_\text{p}$ in a similar way as explained above. This approach is especially tedious when considering many modes because a linear system has to be solved for every possible combination. Our method, on the other hand, computes the LS solution of (8.6) by explicitly exploiting the Kronecker product structure of the coefficients.

8.3 Numerical experiments

We illustrate our KPE-based method for the extended Yale Face Database B[1] which contains cropped facial images of $J = 37$ persons under 64 illumination conditions. Some of the illuminations are missing for several persons and are therefore removed entirely from the dataset, obtaining $I = 57$ conditions. Each image of size 51×58 pixels is vectorized into a vector of length $M = 2958$. Hence, the resulting tensor $\mathcal{D}$ has size $2958 \times 37 \times 57$.

We use a nonlinear LS algorithm with random initialization in order to solve (c)KPEs [26]. All computations are done with Tensorlab [215]. We compute the MLSVD with a randomized algorithm called `mlsvd_rsi` [211]. We use $R = 15$, $L = J$, and $P = 1000 \ll M$ which we determined via cross validation. We project the given image onto the column space of $\mathbf{U}_\text{px}$ in order to reduce computation time. In order to accommodate for scaling and sign invariance, the rows of $\mathbf{U}_\text{p}$ and the estimated coefficient vectors are normalized as $\text{sign}(c_1)\frac{\mathbf{c}}{\|\mathbf{c}\|}$. As explained in subsection 8.2.3, the normalized rows of $\mathbf{U}_\text{p}$ act as database, denoted by $\mathbf{U}_\text{db}$, and the normalized coefficient vector for the person mode acts as a feature vector.

8.3.1 Proof-of-concept

Although the facial image in Figure 8.1 is almost completely dark, our method correctly recognizes the person in the image. In this case, we constructed the MLSVD model in (8.2) using all facial images of all persons under every illumination condition. Hence, the coefficient vectors $\mathbf{c}_p$ and $\mathbf{c}_i$ are perfectly

---

[1] The extended Yale Face Database B can be downloaded from `http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html` (visited March 14, 2018.

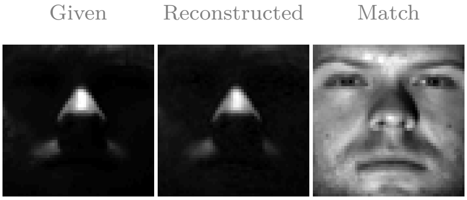


Figure 8.1: Classification of a person that is included in the dataset. *Note that we can identify the person even though the picture is almost completely dark.*

Table 8.1: By reformulating face recognition as a Kronecker product equation, higher performance (%) can be obtained in comparison to conventional techniques such as Eigenfaces as well as the tensor-based approach in [204]. *Lower recognition time (s) is achieved in comparison to the method of [204].*

	Eigenfaces [194]	Vasilescu [204]	KPE
Accuracy	93.3	93.5	**95.7**
Precision	90.6	94.4	**96.6**
Recall	88.4	90.9	**95.8**
Time of PCA/MLSVD	**2.97**	3.29	3.29
Time of recognition	**0.004**	0.135	0.097

reconstructed and a correct match is found. The reconstructed image in Figure 3.6 can be obtained by recomputing the vectorized image using the estimated coefficient vectors.

8.3.2 Performance

Our method obtains higher performance than conventional techniques as we effectively exploit the multilinear structure of facial images by reformulating the recognition task as a KPE. Although our method is slower than the matrix-based method, it is slightly faster than the tensor-based method from [204] for this dataset. In Table 8.1 we report the median performance and time across 50 trials for our method, Eigenfaces, and the tensor-based method from [204]. In particular, we report the accuracy as well as the precision and recall using macro-averaging as explained in [175]. In each trial, 75% of the illumination conditions are selected randomly as training set and 25% as test set for each person. For Eigenfaces, we unfold the data tensor and apply principal component analysis (PCA): we have $\mathbf{D}_{(1)} = \mathbf{B}\mathbf{C}^{\mathrm{T}}$ with $\mathbf{B} \in \mathbb{R}^{M \times T}$ and $\mathbf{C} \in \mathbb{R}^{IJ \times T}$ using $T = J = 37$.

Table 8.2: Higher performance (%) can be achieved by using multiple images under different illuminations. *Our cKPE-based method outperforms Eigenfaces using majority voting.*

	Eigenfaces [194]			cKPE-based method		
# illuminations	1	2	3	1	2	3
Accuracy	92.7	93.3	96.3	**95.8**	**97.1**	**97.3**
Precision	89.8	91.2	97.9	**97.0**	**99.3**	**99.9**
Recall	87.7	87.8	97.5	**96.2**	**99.2**	**99.9**

8.3.3 Improving performance through coupling

More robust recognition can be achieved by using multiple images of the same person under different illuminations. In Table 8.2 we report the median performance across 15 trials when using $Q = \{1, 2, 3\}$ randomly chosen illuminations. We compare our cKPE-based method to Eigenfaces using majority voting. For the latter, we assign the label of the person with the lowest index in the case of a tie. We use the same experiment settings as in subsection 8.3.2 and solve a cKPE with Q randomly chosen illuminations which we repeat 25 times for each trial and each person. Clearly, higher accuracy can be achieved by using multiple images for both approaches. Our cKPE-based method achieves higher accuracy than Eigenfaces.

8.3.4 Updating the database with a new person

Given an MLSVD model, the KPE-based method allows one to update the database $\mathbf{U}_{\text{db}}$ with a new person using *only a few* images instead of an image for each illumination. For example, consider an MLSVD model as in (8.2) which we have constructed using the facial images of all but one person under every illumination. The retained person is initially not included in the database $\mathbf{U}_{\text{db}}$, but can be recognized as follows. By solving (8.6) for a particular image, we obtain a feature vector $\tilde{\mathbf{c}}_p^{(\text{new})}$ which we can add as a new row to the extended database $\mathbf{U}_{\text{db}}$ (with known label). In order to recognize the person in a new image under a different illumination, we can proceed as before, i.e., we solve a KPE and compare the obtained feature vector with the extended database. This strategy works well if the given image can be well approximated by the original MLSVD model. In practice, one can improve the recognition by extending the database using multiple illuminations and solving a cKPE to obtain a new row for the database.

We illustrate the approach for the person depicted in Figure 8.2 (left). In this example, we choose a neutral illumination to update the database. The current MLSVD captures the new person reasonably well as can be seen from the reconstructed image on the right. The person is correctly recognized in a new image with a different illumination as illustrated in Figure 8.3.

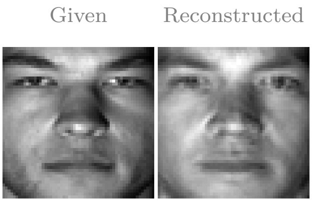


Figure 8.2: The MLSVD model captures the new person reasonably well.

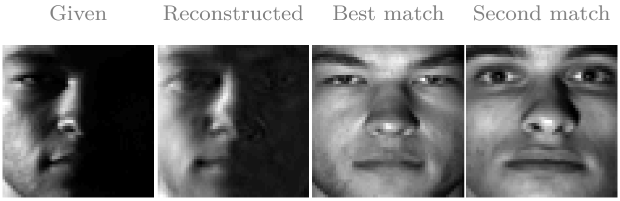


Figure 8.3: Although we update the database with a new person using only one illumination condition, the KPE-based method recognizes that person in a new image under a different illumination condition.

By using multiple illuminations to update the database, one can again further improve the performance. In Table 8.3 we report the performance when updating the database with the first, 16th, or 28th person in the extended Yale Face Database B using all other persons to construct the model. In other words, the data is divided into a training set of 36 persons and a test set of 1 person. In this experiment, we use $P = 1000 \ll M$. When using one, two, or three images, we use illumination setting 1, $\{1, 10\}$, and $\{1, 10, 55\}$, respectively. Illumination 1, 10, and 55 correspond to the neutral illumination and a left and right illumination of the face, respectively. Clearly, the accuracy improves by updating the model using multiple illuminations as can be seen in Table 8.3. Also, one can see that the median performance over all persons in the dataset improves by using additional illuminations.

Table 8.3: When updating the database with a new person, our method can achieve higher accuracy (%) by fusing multiple images under different illumination conditions instead of using only one image of the new person.

Person	One illumination	Two illuminations	Three illuminations
1	58.9	70.9	**77.8**
16	41.1	56.4	**70.4**
28	53.6	54.5	**75.3**
All	50.0	50.9	**64.8**

8.4 Conclusion

In this chapter, we proposed a new tensor-based technique for face recognition that exploits the multidimensional nature of a collection of facial images under different conditions such as illumination, pose, and expression. First, we construct a tensor by stacking the images along several modes that each relate to a variation in the image. Our method models the obtained tensor by a multilinear SVD, describing each of the modes with a factor matrix and the interaction between the modes with a tensor. By reformulating the recognition task as the computation of a KPE, we can explicitly exploit the multilinear structure of the problem, obtaining a feature vector that enables higher performance than conventional methods. We illustrated our method for the extended Yale Face Database B, obtaining better performance than Eigenfaces and another tensor-based technique. Our method performs well when using only a single image and the performance can be improved further by coupling a few images with different illuminations. Remarkably, our method also allows one to update the database with a new person using only a few images instead of an image for each combination of conditions. In future work, one can probably improve the performance by using neural networks or support vector machines in combination with KPE-computed feature vectors instead of using Euclidian distance-based comparisons. Additionally, one can take into account the nonnegative nature of the data.

Irregular heartbeat classification using Kronecker product equations

9

ABSTRACT | Cardiac arrhythmia or irregular heartbeats are an important feature to assess the risk on sudden cardiac death and other cardiac disorders. Automatic classification of irregular heartbeats is therefore an important part of ECG analysis. We propose a tensor-based method for single- and multi-channel irregular heartbeat classification. The method tensorizes the ECG data matrix by segmenting each signal beat-by-beat and then stacking the result into a third-order tensor with dimensions channel × time × heartbeat. We use the multilinear singular value decomposition to model the obtained tensor. Next, we formulate the classification task as the computation of a Kronecker product equation. We apply our method on the INCART dataset, illustrating promising results.

This chapter is a slightly adapted version of M. Boussé, G. Goovaerts, N. Vervliet, *et al.*, "Irregular heartbeat classification using Kronecker product equations", in *39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2017, Jeju Island, South-Korea)*, Jul. 2017, pp. 438–441.

9.1 Introduction

Cardiac arrhythmia or irregular heartbeats are conditions where the behavior of the heart is abnormal. This is characterized by the heart beating either too slow, too fast, or irregularly. In many cases irregular heartbeats do not require medical attention. However, certain types of arrhythmia such as ventricular fibrillation are medical emergencies that may lead to sudden cardiac death. The presence of arrhythmia can also be an indication of cardiac disorders. It is therefore essential that irregular heartbeats can be detected in a reliable way. Also, the rise of online and long-term ECG monitoring has increased the need for automated heartbeat classification methods. When an ECG signal contains thousands of heartbeats, manual beat inspection becomes a time-consuming and tedious task which is prone to human errors. Automatic irregular heartbeat detection methods are therefore an important tool in the diagnosis of patients at risk for cardiac events.

Traditional heartbeat classification methods often use RR interval or ECG morphology features [138], [192]. These methods typically represent the ECG signal as a vector. Recently, there is a trend to represent the signals in multi-lead ECG as a tensor in order to preserve structural information [93], [109], [132]. A tensor is a higher-order generalization of a vector (first-order) and a matrix (second-order). In this chapter, we tensorize the ECG data matrix into a third-order tensor with dimensions channel × time × heartbeat by means of segmentation [93]. This tensorization technique segments the ECG signal of each channel beat-by-beat and stacks the results into a third-order tensor, enabling the use of tensor decompositions.

We propose a new tensor-based method for irregular heartbeat classification which can classify new heartbeats as regular or irregular using the ECG signal of a single channel. First, we model the obtained tensor using a multilinear singular value decomposition (MLSVD) [54]. We show that every heartbeat in the tensor can then be expressed as a Kronecker product equation (KPE). The latter is a linear system of equations with a Kronecker product structured solution [26]. In order to classify a new heartbeat signal with an unknown label, we solve a similar KPE which allows us to find the closest match with a labeled heartbeat in the tensor. In practice, the MLSVD model is only approximate and robustness can be improved by using several channels instead of just one, leading to a coupled KPE. We illustrate our method on the INCART dataset.

In the remainder of this section we introduce the notation and basic definitions as well as the MLSVD and (coupled) KPEs. We present our method in Section 9.2 and discuss experiments in Section 9.3. We conclude the chapter in Section 9.4.

9.1.1 Notation and definitions

We denote vectors, matrices, and tensors by bold lower (e.g., $\mathbf{a}$), bold uppercase (e.g., $\mathbf{A}$), and calligraphic letters (e.g., $\mathcal{A}$), respectively. The nth element in a sequence is indicated by a superscript between parentheses, e.g., $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$. A mode-$n$ vector of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined by fixing every index except the nth and is a natural extension of the rows and columns of a matrix. The mode-n unfolding of $\mathcal{A}$ is a matrix $\mathbf{A}_{(n)}$ with mode-n vectors as its columns (following the ordering convention in [125]). The vectorization of $\mathcal{A}$, denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \cdots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1 + \sum_{k=1}^{N}(i_k - 1)J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$.

The outer and Kronecker product are denoted by $\circledcirc$ and $\otimes$, respectively, and are related by $\operatorname{vec}(\mathbf{a} \circledcirc \mathbf{b}) = \mathbf{b} \otimes \mathbf{a}$. The mode-$n$ product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $\mathbf{B} \in \mathbb{R}^{J_n \times I_n}$ is a tensor $\mathcal{A} \cdot_n \mathbf{B} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots I_N}$ and is defined element-wise as

$$(\mathcal{A} \cdot_n \mathbf{B})_{i_1 \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} a_{i_1 i_2 \cdots i_N} b_{j_n i_n}$$

Hence, each mode-n vector of the tensor $\mathcal{A}$ is multiplied with $\mathbf{B}$, i.e., $(\mathcal{A} \cdot_n \mathbf{B})_{(n)} = \mathbf{B}\mathbf{A}_{(n)}$.

An Nth-order tensor of rank one is defined as the outer product of N nonzero vectors [168]. The rank of a tensor equals the minimal number of rank-1 tensors that generate it as their sum. The mode-n rank of a tensor is defined as the rank of the mode-n unfolding of the tensor. The multilinear rank of an Nth order tensor is equal to the N-tuple of mode-n ranks.

9.1.2 Multilinear singular value decomposition

The multilinear singular value decomposition (MLSVD) of a higher-order tensor is a multilinear generalization of the singular value decomposition (SVD) of a matrix [41], [54], [168].

Definition 29. A *multilinear singular value decomposition* (MLSVD) writes a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ as the product

$$\mathcal{A} = \mathcal{S} \cdot_1 \mathbf{U}^{(1)} \cdot_2 \mathbf{U}^{(2)} \cdots \cdot_n \mathbf{U}^{(N)},$$

in which $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ is a unitary matrix, $1 \leq n \leq N$, and the core $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is ordered and all-orthogonal.

The MLSVD is a powerful tensor tool in applications such as compression and dimensionality reduction [59], [125]. It is related to the low-multilinear rank approximation (LMLRA) and the Tucker model, see [54], [214] and references therein.

9.1.3 Kronecker product equations

A KPE is a linear system of equations with a solution that has a Kronecker product structure [26]. Consider a system $\mathbf{A}\mathbf{x} = \mathbf{b}$ with $\mathbf{A} \in \mathbb{R}^{M \times K}$, $\mathbf{x} \in \mathbb{R}^{K}$, and $\mathbf{b} \in \mathbb{R}^{M}$. Assume the solution $\mathbf{x}$ is constrained to the following simple Kronecker product structure: $\mathbf{x} = \mathbf{v} \otimes \mathbf{u}$ with $\mathbf{u} \in \mathbb{R}^{I}$ and $\mathbf{v} \in \mathbb{R}^{J}$ such that $K = IJ$. As such, we have that:

$$\mathbf{A}(\mathbf{v} \otimes \mathbf{u}) = \mathbf{b}. \tag{9.1}$$

The Kronecker product structure can be exploited in order to rewrite (9.1) as a multilinear system of equations [26]:

$$\mathcal{A} \cdot_2 \mathbf{u}^{\mathrm{T}} \cdot_3 \mathbf{v}^{\mathrm{T}} = \mathbf{b}$$

with the coefficient tensor $\mathcal{A} \in \mathbb{R}^{M \times I \times J}$ defined such that its mode-1 unfolding $\mathbf{A}_{(1)} \in \mathbb{R}^{M \times IJ}$ equals the coefficient matrix $\mathbf{A}$ in (9.1), i.e., we have that $\mathbf{A}_{(1)} = \mathbf{A}$.

A coupled KPE (cKPE) is a set of KPEs that share a coefficient vector. We limit ourselves to cKPEs of the form:

$$\mathbf{A}(\mathbf{v}^{(q)} \otimes \mathbf{u}) = \mathbf{b}^{(q)} \text{ for } 1 \leq q \leq Q \tag{9.2}$$

with $\mathbf{A} \in \mathbb{R}^{M \times K}$, $\mathbf{v}^{(q)} \in \mathbb{R}^{I}$, $\mathbf{u} \in \mathbb{R}^{J}$, and $\mathbf{b}^{(q)} \in \mathbb{R}^{M}$ such that $K = IJ$. We can reformulate (9.2) as a multilinear system:

$$\mathcal{A} \cdot_2 \mathbf{u}^{\mathrm{T}} \cdot_3 \mathbf{V}^{\mathrm{T}} = \mathbf{B} \tag{9.3}$$

in which $\mathbf{V} \in \mathbb{R}^{I \times Q}$ with $\mathbf{v}_q = \mathbf{v}^{(q)}$, $\mathbf{B} \in \mathbb{R}^{M \times Q}$ with $\mathbf{b}_q = \mathbf{b}^{(q)}$, and $\mathbf{A}_{(1)} = \mathbf{A}$. Expression (9.3) is equivalent with:

$$\mathbf{A}(\mathbf{V} \otimes \mathbf{u}) = \mathbf{B}$$

which can be interpreted as a more general type of KPE.

9.2 Irregular heartbeat classification as a Kronecker product equation

9.2.1 Preprocessing and tensorization

The preprocessing step is necessary to remove noise from the ECG signal that may corrupt the final classification performance. Similarly as in [93], we consider baseline wander and high frequency noise from muscle artifacts as primary noise sources. They are removed channel-by-channel using quadratic variation reduction and wavelet-based filtering, respectively.

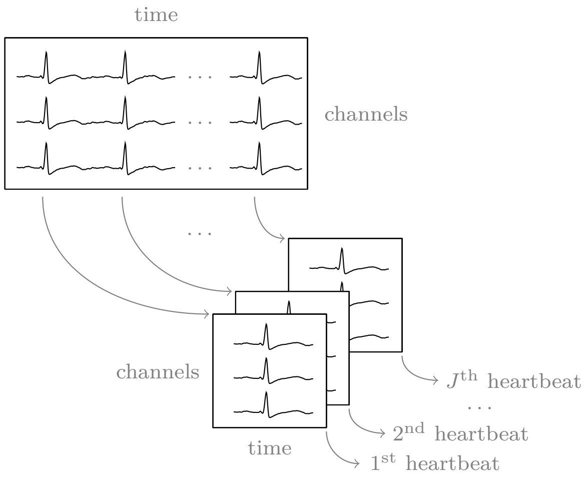

time
channels
...
channels
time
J^{th} heartbeat
...
2^{nd} heartbeat
1^{st} heartbeat

Figure 9.1: Tensorization of an ECG data matrix into a third-order tensor with dimensions channel × time × heartbeat using segmentation.

Next, we transform the ECG data matrix into a third-order tensor as illustrated in Figure 9.1. First, we segment the signals into smaller segments of size I containing only a single heartbeat. As such, we obtain J heartbeats for all M channels. Next, we stack all heartbeats in the third mode, obtaining a third-order tensor $\mathcal{T} \in \mathbb{R}^{M \times I \times J}$ with dimensions channel × time × heartbeat. We use this particular tensorization because we are only interested in the differences between subsequent heartbeats but other techniques can be found in literature [62].

Segmentation in individual heartbeats is done here by taking a fixed-size window of 500 ms around each R peak, starting 200 ms before the peak. The R peak location can easily be detected using standard techniques such as Pan-Tompkins. Note that when the heart rate changes a lot throughout the signal (for example in long-term ambulatory signals), resampling the heartbeats might be required to align the different ECG waves.

9.2.2 Kronecker product equation

The (truncated) MLSVD of the tensor $\mathcal{T}$ is given by:

$$\mathcal{T} = \mathcal{S} \cdot_1 \mathbf{U}_{\text{c}} \cdot_2 \mathbf{U}_{\text{t}} \cdot_3 \mathbf{U}_{\text{h}} \tag{9.4}$$

with $\mathbf{U}_{\text{c}} \in \mathbb{R}^{M \times P}$, $\mathbf{U}_{\text{t}} \in \mathbb{R}^{I \times R}$, and $\mathbf{U}_{\text{h}} \in \mathbb{R}^{J \times L}$ forming an orthonormal basis for the spatial, temporal, and shape component, respectively. The coefficient tensor $\mathcal{S} \in \mathbb{R}^{P \times R \times L}$ explains the interaction between the different modes. *Every* heartbeat $\mathbf{t} \in \mathbb{R}^{I}$ of a *particular* channel, i.e., every mode-2 vector of the tensor $\mathcal{T}$, satisfies the following model:

$$\mathbf{t}^{\mathrm{T}} = \mathcal{S} \cdot_1 \mathbf{c}_{\mathrm{c}}^{\mathrm{T}} \cdot_2 \mathbf{U}_{\mathrm{t}} \cdot_3 \mathbf{c}_{\mathrm{h}}^{\mathrm{T}}. \tag{9.5}$$

Vectors $\mathbf{c}_{\mathrm{c}}^{\mathrm{T}}$ and $\mathbf{c}_{\mathrm{h}}^{\mathrm{T}}$ are rows of $\mathbf{U}_{\mathrm{c}}$ and $\mathbf{U}_{\mathrm{h}}$, respectively, corresponding to the coefficients of heartbeat h and channel c.

Clearly, the mode-2 unfolding of (9.5) is a KPE:

$$\mathbf{t} = \mathbf{U}_{\mathrm{t}} \mathbf{S}_{(2)} (\mathbf{c}_{\mathrm{h}} \otimes \mathbf{c}_{\mathrm{c}}). \tag{9.6}$$

Equation (9.6) expresses $\mathbf{t}$ in the column space $\mathbf{U}_{\mathrm{t}}$ and (the mode-2 unfolding of) an additional interaction tensor $\mathcal{S}$ that links the different modes. The coefficients can then be written as a Kronecker product of the coefficient vectors $\mathbf{c}_{\mathrm{h}}$ and $\mathbf{c}_{\mathrm{c}}$.

We can also consider a set of K channels instead of just one. In that case we have a set of K KPEs that are coupled via the coefficient vector for the heartbeat dimension:

$$\mathbf{t}^{(q)} = \mathbf{U}_{\mathrm{t}} \mathbf{S}_{(2)} (\mathbf{c}_{\mathrm{h}} \otimes \mathbf{c}_{\mathrm{c}}^{(q)}) \text{ for } 1 \leq q \leq Q.$$

We collect all heartbeat signals $\mathbf{t}^{(q)}$ in $\mathbf{T} \in \mathbb{R}^{I \times Q}$ and all channel coefficients $\mathbf{c}_{\mathrm{c}}^{(q)}$ in $\mathbf{C}_{\mathrm{c}} \in \mathbb{R}^{M \times K}$. As such, we obtain:

$$\mathbf{T}^{\mathrm{T}} = \mathcal{S} \cdot_1 \mathbf{C}_{\mathrm{c}}^{\mathrm{T}} \cdot_2 \mathbf{U}_{\mathrm{t}} \cdot_3 \mathbf{c}_{\mathrm{h}}^{\mathrm{T}}$$

which is equivalent with the following cKPE:

$$\mathbf{T} = \mathbf{U}_{\mathrm{t}} \mathbf{S}_{(2)} (\mathbf{c}_{\mathrm{h}} \otimes \mathbf{C}_{\mathrm{c}}). \tag{9.7}$$

9.2.3 Irregular heartbeat classification

We explain how to classify a *new* heartbeat measured on a *single* channel as regular or irregular using KPEs. Consider an ECG data matrix with known heartbeat labels. First, we perform preprocessing and tensorization as explained in Subsection 9.2.1, obtaining a tensor $\mathcal{T}$. Next, we compute a MLSVD of $\mathcal{T}$ as in (9.4), obtaining factor matrices $\mathbf{U}_{\mathrm{c}}$, $\mathbf{U}_{\mathrm{t}}$, and $\mathbf{U}_{\mathrm{h}}$ and core tensor $\mathcal{S}$. Recall that every heartbeat in $\mathcal{T}$ can be expressed as a KPE as in (9.6). Consider now a *new* heartbeat $\mathbf{t}^{(\mathrm{new})}$ with unknown label, i.e., a heartbeat that is not included in $\mathcal{T}$. In order to classify the new heartbeat, we solve a KPE:

$$\mathbf{U}_{\mathrm{t}} \mathbf{S}_{(2)} (\mathbf{c}_{\mathrm{h}}^{(\mathrm{new})} \otimes \mathbf{c}_{\mathrm{c}}^{(\mathrm{new})}) = \mathbf{t}^{(\mathrm{new})},$$

obtaining estimates $\tilde{\mathbf{c}}_{\mathrm{h}}^{(\mathrm{new})}$ and $\tilde{\mathbf{c}}_{\mathrm{c}}^{(\mathrm{new})}$ for the unknown coefficient vectors $\mathbf{c}_{\mathrm{h}}^{(\mathrm{new})}$ and $\mathbf{c}_{\mathrm{c}}^{(\mathrm{new})}$, respectively. We compare $\tilde{\mathbf{c}}_{\mathrm{h}}^{(\mathrm{new})}$ with the rows of $\mathbf{U}_{\mathrm{h}}$ using the norm of the difference (after fixing scaling and sign invariance). We then classify the new heartbeat with the label corresponding to the closest match.

We use the data of all channels to compute the MLSVD but classify using the signal from a single channel. In practice, however, the MLSVD model holds only approximately and incorrect classification can possibly occur. We can make the classification more robust by using heartbeats from *multiple* channels which can be solved using a coupled KPE as in (9.7).

9.3 Results and discussion

We illustrate the proposed method with two experiments using the first ten subjects of the St.-Petersburg Institute of Cardiological Technics 12-lead Arrhythmia (INCART) Database available on Physionet [91]. The dataset consists of 75 ECG recordings from 32 subjects. All signals are 30 minutes long and contain 12 standard leads. The sampling frequency is 257 Hz. The signals are collected during tests for coronary artery diseases. The dataset contains all ECG signals together with patient diagnoses, R peak locations and beat annotations. The beat annotations were first automatically determined and later corrected manually. We apply preprocessing and segmentation as explained in Subsection 9.2.1 and obtain heartbeats of length $I = 131$. The number of heartbeats J is different for each subject. The number of channels is $M = 12$.

We developed nonlinear least-squares (NLS) algorithms for solving KPEs and cKPEs, called `kpe_nls` and `ckpe_nls`, respectively, which are available upon request [26]. All computations are done with Tensorlab [215]. We compute the MLSVD with a randomized algorithm called `mlsvd_rsi` which is faster but achieves similar accuracy than non-randomized MLSVD algorithms [211]. We use $P = M = 12$ and $R = I = 131$. Strongly truncating the third mode, i.e., taking $L \ll J$, decreases computation time and improves classification performance. The optimal value for L is subject dependent and can be determined via validation data with $2 \leq L \leq 10$. We use random initialization in all experiments. Each row of $\mathbf{U}_{\mathrm{h}}$ and the estimated coefficient vectors are normalized to accommodate for scaling and sign invariance as follows: a vector $\mathbf{c}$ is normalized to $\bar{\mathbf{c}}$ as $\bar{\mathbf{c}} = \mathrm{sign}(c_1)\frac{\mathbf{c}}{\|\mathbf{c}\|}$.

In a first experiment, we show that our method achieves high classification performance provided we choose a suitable channel for classification. This is illustrated in Figure 9.2 where we report the median across 30 trials of the sensitivity and specificity for subjects one and four and all channels. The data for subject one and four consist of 2411 and 2301 regular and 344 and 121 irregular heartbeats, respectively [91]. For each subject, we randomly divided the data in training (85%) and (15%) test set in each trial. We used $L = 4$ and $L = 8$ in the MLSVD model, respectively. Clearly, the performance depends on the choice of the channel and the choice is subject dependent. For example, the highest specificity for subject one (0.8173) and four (0.8173) is achieved if

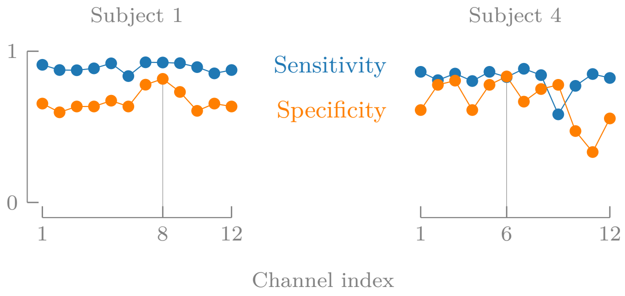


Figure 9.2: Overall our method achieves good performance while better results can be obtained by using a suitable channel for a given subject, e.g., channels 8 and 6 achieve the highest specificity for subjects 1 and 4, respectively.

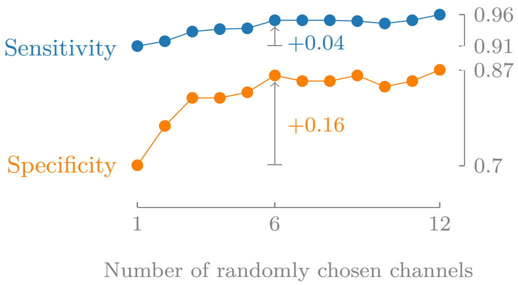


Figure 9.3: Fusing signals from multiple channels leads to a better performance.

one uses channel eight (V2) and six (AVF), respectively. However, the overall performance is also good: the median sensitivity and specificity across all subjects, using, e.g., channel eight (V2), is 0.9083 and 0.7353, respectively. Moreover, in that case the F_1 score is 94.2% which is better than the best performance (92%) of traditional techniques as in [138] that use all channels. It is remarkable that our method can achieve high performance while using only a single channel for classification.

Fusing the ECG signals from multiple channels with our method improves classification performance. In Figure 9.3 we report the median across 10 trials of the sensitivity and specificity for subject one (using $L = 4$). In each trial we also randomly divided the data in training (85%) and test (15%) set. The number of channels that is used for classification is varied from one to twelve and in each trial the channels are chosen randomly. For example, coupling six random channels greatly improves the specificity for subject one. However, only a small improvement is obtained for the sensitivity. Also, coupling more than six channels does not seem to increase the overall performance significantly for this subject.

9.4 Conclusion

We presented a new tensor-based method for single- and multi-channel irregular heartbeat classification. The proposed method tensorizes the ECG data matrix using segmentation. The obtained tensor is modeled by a MLSVD which allows us to express every heartbeat in the tensor as a KPE. We have shown that the classification task can then be formulated as the computation of a KPE. While the method performs well for only a single channel, the performance can be improved by coupling the ECG signals from multiple channels by means of a cKPE. We illustrated our method on the INCART dataset. The proposed method can achieve high performance by choosing a suitable channel for classification. Coupling multiple channels, improved the overall classification performance. In future work, the method can be extended to multi-class classification. Also, one can possibly improve the performance by using more intricate schemes to determine the best match in the database. Finally, further research is necessary to determine the best channel(s) to use for classification in both the single- and multi-channel case.

Conclusion

10

10.1 Contributions

We give a chapter-by-chapter overview of our contributions. For chapters that rely on results obtained in collaboration with other team members, we clearly indicate the contribution of each collaborator.

Chapter 3

In this chapter, we report the results of a collaboration with N. Vervliet, I. Domanov, and O. Debals. The doctoral candidate has contributed to the formulation of the LS-CPD concept, the optimization-based algorithms, and the face recognition application.

- *Implicit tensor decompositions.* LS-CPDs can be interpreted as the computation of a CPD of a tensor that is only known *implicitly* via the solution of a linear system of equations. By *directly* solving the problem via optimization-based and algebraic methods, instead of first solving the unstructured system and then decomposing the reshaped solution, we can obtain a unique solution in the underdetermined case.

- *Multilinear systems of equations.* LS-CPDs can be seen as multilinear systems of equations, which are a generalization of linear systems of equations. This is analogous to tensor decompositions being higher-order generalizations of matrix decompositions. However, in contrast to tensor decompositions, multilinear systems of equations have not been extensively studied. LS-CPDs provide an initial framework to solve and analyze multilinear systems of equations.

- *Generic uniqueness conditions.* Consider a linear system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$ with $\mathbf{x}$ a vectorized CPD with random factor matrices. For a generic matrix $\mathbf{A}$, the CPD can be recovered uniquely with probability one if the number of equations is strictly larger than the number of free variables in the CPD. This result was contributed by I. Domanov.

- *Algebraic algorithm for rank-1 tensors.* Given a linear system $\mathbf{Ax} = \mathbf{b}$ with $\mathbf{x}$ a vectorized rank-1 tensor, we derived an algorithm to compute the CPD even if $\mathbf{A}$ does not have full column rank, provided some conditions are satisfied. This result was contributed by I. Domanov.

- *Optimization-based algorithm.* Instead of first solving the linear system without structure and subsequently decomposing a tensorized version of the obtained solution, we proposed a GN-style method that solves the LS-CPD problem *directly*, allowing us to work more efficiently and avoid error accumulation. By exploiting all available structure in $\mathbf{A}$, the computational complexity of the generic algorithm can be reduced as we have illustrated for sparse $\mathbf{A}$ and Kronecker structure, see Chapter 4. The rank-1 case was contributed by N. Vervliet and the extension to $R > 1$ was provided by the doctoral candidate.

- *Tensor-based face recognition.* LS-CPDs can be used for tensor-based classification tasks which is illustrated in this chapter for tensor-based face recognition. We have shown that the recognition rate can be improved by using the LS-CPD approach instead of TensorFaces [202]. See also Chapter 8 and Chapter 9.

- *Algorithm for constructing tensors with particular multilinear singular values.* While the construction of a matrix with particular singular values is trivial, the problem is not straightforward for tensors. We have shown that the problem can in fact be formulated as a LS-CPD by writing the orthogonality constraints and the constraints imposing multilinear singular values as a linear system with a Kronecker-product constrained solution. By exploiting the sparsity of the resulting matrix $\mathbf{A}$, one can obtain a more efficient algorithm with respect to existing techniques. The problem was formulated by I. Domanov and the sparsity-exploiting implementation was developed by N. Vervliet.

- *Blind deconvolution of constant-modulus signals.* We have shown that the blind deconvolution of constant modulus signals can be reformulated as the computation of a LS-CPD. Our *generic* framework obtains similar or better accuracy as state-of-the-art techniques for comparable run times. The application was contributed by O. Debals.

Chapter 4

- *Kronecker-structured LS-CPD.* A GN-based algorithm for Kronecker-structured LS-CPD is derived. By fully exploiting all available structure of the measurement matrix, the computational complexity of the dedicated algorithm can be significantly reduced, allowing us to tackle large-scale problems.

- *Graph clustering.* We have shown that a similarity-based clustering method can be reformulated as the computation of a Kronecker-structured LS-CPD. In contrast to existing methods, our approach can easily be extended to higher-order approximations.

Chapter 5

In this chapter, we partially report results originating from the master's thesis by the doctoral candidate under daily supervision of O. Debals.

- *Large-scale segmentation-based blind source separation.* We derived a deterministic method for (large-scale) BSS that exploits the fact that signals in large-scale problems often admit a compact representation. By using *segmentation* and low-rank matrix and tensor representations to model the sources and/or mixing vectors, BSS can be reformulated as the computation of a tensor decomposition, enabling a unique solution because of the mild uniqueness properties in the higher-order setting. In contrast to existing methods, we impose only mild conditions on the sources via the uniqueness conditions, e.g., linear independence instead of statistical independence as in ICA. By employing tensors of higher order, larger compression rates can be achieved, making our approach feasible for large-scale problems.

- *Connection between Hankelization and segmentation.* Segmentation is a deterministic tensorization technique that is closely related to Hankel-based tensorization. The matrix obtained via segmentation is a subset of the columns of the Hankel matrix and therefore low-rank Hankel matrices also yield low-rank matrices after segmentation.

- *Flower and butterfly decomposition.* Higher-order segmentation and two-fold segmentation result into the flower and butterfly decomposition, respectively, which are two novel decompositions. An algebraic algorithm and the uniqueness conditions for the flower decomposition are provided in the follow-up paper presented in Chapter 6.

- *Fetal ECG extraction.* By exploiting the intrinsic low-rank structure of the QRS complexes in the ECG signal, our method can achieve a clear separation of the fetal and maternal ECG from multilead cutaneous potential recordings of pregnant women.

- *Direction-of-arrival estimation.* Estimating the direction of arrival (DOA) of signals is an important application in telecommunication and array processing. By exploiting the low-rank structure of the mixing vectors in ULAs, one can extract the DOA parameters. Thanks to the compact representation, our approach scales well to large-scale problems. Besides the far-field setting, our method can also tackle the near-field and multipath settings well.

Chapter 6

In this chapter, we partially report results originating from the master's thesis by the doctoral candidate under daily supervision of O. Debals.

- *Large-scale segmentation-based blind system identification.* We successfully generalized our segmentation-based approach for (large-scale) BSS to the blind identification of large-scale convolutive systems. More specifically, we applied the same idea to the system coefficients of FIR models, allowing us to reformulate the identification problem as the computation of a structured tensor decomposition. The fairly mild uniqueness conditions in the higher-order setting allow us to obtain a unique solution.

- *Flower decomposition.* By applying segmentation to the BSI problem, we obtained a flower decomposition for which we have provided improved uniqueness conditions and an algebraic algorithm that can be used to effectively initialize optimization-based methods.

- *Low-rank representation of periodic signals.* We have shown that general periodic signals can be represented by low-rank matrices, even in cases where the period is unknown or may have been estimated inaccurately. We proved this fact by connecting periodic signals to circulant matrices in a similar fashion as exponential signals are linked to Hankel matrices.

- *Improved uniqueness conditions.* We have provided improved uniqueness conditions for the Toeplitz-constrained flower decomposition by explicitly exploiting the Toeplitz structure.

- *Parameter analysis.* We performed a detailed analysis of the segmentation parameters, resulting into a set of practical guidelines for applying segmentation.

- *Direction-of-arrival estimation.* We have shown that our method outperforms existing methods such as 2D-MUSIC for large-scale uniform rectangular arrays (URA). Our method also proves viable for DOA estimation in large-scale URAs with possibly broken antennas and even in non-uniform arrays.

- *Neural spike sorting.* We have shown that our segmentation-based approach can obtain an excellent separation of a convolutive mixture of (simulated) spike trains stemming from high-density microelectrode arrays for measuring neuronal activity.

Chapter 7

- *Large-scale AR system identification.* By explicitly exploiting the hypothesized structure of the system coefficients in large-scale applications, we have shown that the identification problem can be reformulated as the computation of a KPE, allowing us to use optimization-based solvers and uniqueness conditions.

- *Explicit versus implicit tensor decompositions and a link with FIR and AR models.* Applying our segmentation-based approach to FIR models results into *explicit* tensor decomposition-based algorithms, as shown in Chapter 6, while applying the approach to AR models results into *implicit* tensor decomposition-based algorithms such as the LS-CPD framework of Chapter 3.

Chapter 8

In this chapter, we report the results of a collaboration with N. Vervliet and O. Debals.

- *Improved recognition rate via KPEs.* Our KPE-based method is more efficient and more accurate than TensorFaces [202] for tensor-based face recognition. In contrast to TensorFaces, our method can also be used to add an unknown person to the database using only one (or a few) images instead of an image for each combination of conditions. The performance is illustrated for the Extended Yale B dataset.

- *Improved robustness via coupled KPEs.* By using multiple images of a person under different illuminations, one can further improve the performance. The problem can then be formulated as the computation of a *coupled* KPE which is a generalization of the LS-CPD algorithm from Chapter 3.

Chapter 9

In this chapter, we report the results of a collaboration with G. Goovaerts.

- *Single-lead irregular heartbeat classification.* Our KPE-based method can achieve high performance provided we choose a suitable lead for classification. In contrast to existing methods that use all leads for classification, our method uses only a single channel, providing interesting possibilities for wearable applications. The performance is illustrated for the INCART dataset.

- *Improved robustness by using multiple leads.* While our method performs well for only a single channel, the performance can be improved by coupling the ECG signals from multiple leads by means of coupled KPEs. See also Chapter 8.

Appendix A

- *Low-rank weights.* A GN-style method for the computation of a CPD using low-rank weights has been proposed, allowing us to explicitly accommodate for general (co)variances in the least-squares cost function. By using a PD model for the weight tensor, efficient expressions for the optimization ingredients are derived.

10.2 Perspectives

We suggest the following *major* future directions:

- *Extension to other tensor decompositions.* Although the focus of this thesis was on CPD-constrained explicit and implicit tensor decompositions, our approach can also be extended to other decompositions such as the MLSVD, TT, or HT models. In tensor-based scientific computing, one often uses TT and HT models because they combine large compression rates with good numerical properties.

- *Tensor-based learning.* The LS-CPD framework enables tensor-based pattern recognition, data analysis and learning: multilinear regression and classification tasks can be reformulated as *structured* LS-CPD problems. By carefully exploiting all available structure, efficient alternatives to nonlinear machine learning techniques can be derived.

- *Multilinear systems of equations.* LS-CPD problems can be interpreted as multilinear systems of equations. While the latter are a higher-order generalization of linear systems of equations, tensor decompositions are a higher-order extension of matrix decompositions. However, in contrast to tensor decompositions, the domain of multilinear systems is relatively unexplored; only a few cases have been studied in a disparate manner in the literature. Various techniques and applications for multilinear system identification can be explored.

- *Large-scale system identification techniques.* Our KPE-based method for SIMO AR system identification can be extended to the MIMO case by means of the LS-CPD framework. Additionally, the block-Toeplitz structure can be exploited in order to improve the computational complexity and the uniqueness conditions. Furthermore, identifying large-scale autoregressive-moving average (ARMA) and state-space models is an obvious extension of our segmentation-based methods for FIR and

AR models, enabling more general system-modeling techniques. Recently, methods have been proposed for large-scale AR and state-space identification using a similar philosophy as our segmentation-based approach, but limited to a second-order low-rank model [143], [173], [174]. A similar philosophy has also been employed recently for systems with many delays [69], [82], [156].

We also propose several *minor* suggestions:

- *Segmentation versus Hankelization.* Segmentation provides a maximally compact representation of the data, while Hankelization maximally exploits shift invariance. A trade-off between accuracy and compression ratio is possible by stacking partially overlapping segments, allowing us to accommodate to the needs in a particular application.

- *Structured LS-CPD problems.* In order to obtain a computationally efficient algorithm for structured LS-CPD problems, it is important to exploit all available structure. In this thesis, we have focused on sparsity and the Kronecker format, but other structure can be investigated such as Toeplitz or banded-matrix structures.

Nonlinear least-squares algorithm for canonical polyadic decomposition using low-rank weights

A

ABSTRACT | The canonical polyadic decomposition (CPD) is an important tensor tool in signal processing with various applications in blind source separation and sensor array processing. Many algorithms have been developed for the computation of a CPD using a least-squares cost function. Standard least-squares methods assumes that the residuals are uncorrelated and have equal variances which is often not true in practice, rendering the approach suboptimal. Weighted least squares allows one to explicitly accommodate for general (co)variances in the cost function. In this appendix, we develop a new nonlinear least-squares algorithm for the computation of a CPD using low-rank weights which enables efficient weighting of the residuals. We briefly illustrate our algorithm for direction-of-arrival estimation using an array of sensors with varying quality.

This chapter is a slightly adapted version of M. Boussé and L. De Lathauwer, "Nonlinear least squares algorithm for canonical polyadic decomposition using low-rank weights", in *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2017, Curaçao, Dutch Antilles)*, Dec. 2017, pp. 39–43.

A.1 Introduction

Tensors are higher-order generalizations of vectors (first-order) and matrices (second-order). Recently, tensor tools have gained popularity in various applications within signal processing, data mining, and machine learning [41], [125], [168]. An important tensor tool is the canonical polyadic decomposition (CPD) which decomposes a tensor into a minimal sum of rank-1 tensors. The decomposition is unique under mild uniqueness conditions [71], [73], [75], making the CPD a powerful tool in various signal processing applications such as (large-scale) instantaneous and convolutive blind source separation [20], [21], [45], [51], sensor array processing [141], [167], and telecommunications [53]. Various algorithms have been developed for the computation of a CPD such as algebraic methods [47], [72], [75], alternating-least squares (ALS) methods [125], and all-at-once optimization techniques [4], [154], [158], [180], [189]. Although ALS-based methods are popular, they are often outperformed in ill-conditioned cases by more sophisticated optimization techniques such as quasi-Newton (qN) and nonlinear least-squares (NLS) algorithms [180].

In signal processing, the data is often perturbed by noise. Weighted least squares (WLS) allows one to include prior knowledge about the noise in the least-squares cost function. A common choice is the inverse of the sample covariance matrix because it leads to the optimal estimate. In practice, the sample covariance matrix may be unknown but often some reasonable estimate can be computed by exploiting prior knowledge [144]. For example, in array processing the measurements may be observed by sensors with varying quality. In that case, the accuracy of the sensors is often known or can be estimated, allowing one to use weights that are inversely proportional to the variance of the error. Several WLS algorithms have been developed for the CPD such as an all-at-once optimization-based method [154] and a weighted ALS-based approach [106].

In this appendix, we develop a WLS-based algorithm for the computation of a CPD with a weight tensor that can be modeled by a PD, enabling efficient weighting of the residuals. Note that standard least squares corresponds to a rank-1 CPD with factor vectors containing only ones. Additionally, the low-rank structure is interesting for large-scale applications because the CPD provides a compact model for the weights. The CPD structure of the weight tensor allows us to derive efficient expressions for the classical ingredients of standard qN and NLS algorithms. Special care is taken to explicitly exploit the low-rank structure in the derivation. More specifically, we focus on the implementation of a particular type of NLS algorithm, but the expressions can be used for other NLS as well as qN algorithms. We implement the algorithm using the complex optimization framework of Tensorlab, a toolbox for tensor computations in Matlab, as numerical optimization solver [177], [179], [211], [215]. Finally, the WLS-based method with CPD constrained weight tensor is illustrated for direction-of-arrival (DOA) estimation in uniform linear arrays (ULAs) with sensors of varying quality, illustrating excellent results.

We conclude this section with an overview of the notation and basic definitions. Next, we derive efficient expressions for the ingredients of standard qN and NLS methods. WLS for DOA-estimation is illustrated in Appendix A.3.

A.1.1 Notations and basic definitions

Vectors, matrices, and tensors are denoted by bold lowercase, bold uppercase, and calligraphic letters, respectively. The mode-n vector of $\mathcal{A} \in \mathbb{K}^{I_1 \times \cdots \times I_N}$ is a natural extension of the rows and columns of a matrix and is defined by fixing every index except the nth ($\mathbb{K}$ means $\mathbb{R}$ or $\mathbb{C}$). A mode-n unfolding of $\mathcal{A}$ is a matrix $\mathbf{A}_{(n)}$ with the mode-n vectors as its columns (following the ordering convention in [125]). The vectorization of $\mathcal{A}$, denoted as $\operatorname{vec}(\mathcal{A})$, maps each element $a_{i_1 i_2 \cdots i_N}$ onto $\operatorname{vec}(\mathcal{A})_j$ with $j = 1 + \sum_{k=1}^{N}(i_k - 1)J_k$ and $J_k = \prod_{m=1}^{k-1} I_m$.

We indicate the nth element in a sequence by a superscript between parentheses, e.g., $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$. The identity matrix of size $I \times I$ is denoted by $\mathbf{I}_I$. $\mathbf{A} = \operatorname{diag}(\mathbf{a})$ is a diagonal matrix with the entries of $\mathbf{a}$ on the main diagonal. The outer, Kronecker, Hadamard, column-, and row-wise Khatri–Rao product are denoted by $\circledcirc$, $\otimes$, $*$, $\odot$, $\odot^{\mathrm{T}}$, respectively.

The rank of a tensor equals the minimal number of rank-1 tensors that generate the tensor as their sum. A rank-1 tensor is defined as the outer product of nonzero vectors.

A.1.2 Canonical polyadic decomposition

The CPD is an important tensor tool in many applications within signal processing, biomedical sciences, data mining and machine learning; see [41], [125], [168]. The decomposition is unique under rather mild conditions [72], [75] which is a powerful advantage of tensors over matrices in many applications [168].

Definition 30. A *polyadic decomposition* (PD) writes an Nth-order tensor $\mathcal{A} \in \mathbb{K}^{I_1 \times I_2 \times \cdots \times I_N}$ as a sum of R rank-1 terms:

$$\mathcal{A} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circledcirc \mathbf{u}_r^{(2)} \circledcirc \cdots \circledcirc \mathbf{u}_r^{(N)} = \left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)} \right]\!\right].$$

The columns of the factor matrices $\mathbf{U}^{(n)} \in \mathbb{K}^{I_n \times R}$ are equal to the factor vectors $\mathbf{u}_r^{(n)}$ for $1 \leq r \leq R$. The PD is called *canonical* (CPD) when R is equal to the rank of $\mathcal{A}$.

A.1.3 Identities and derivatives

In this appendix, we use the following identities [136], [212]:

$$\left(\mathbf{A}\odot\mathbf{B}\right)^{\mathrm{T}}\left(\mathbf{C}\odot\mathbf{D}\right) = \mathbf{A}^{\mathrm{T}}\mathbf{C} * \mathbf{B}^{\mathrm{T}}\mathbf{D}, \tag{A.1}$$

$$\left(\mathbf{A}\otimes\mathbf{B}\right)\left(\mathbf{C}\otimes\mathbf{D}\right) = \mathbf{AC}\otimes\mathbf{BD}, \tag{A.2}$$

$$\left(\mathbf{A}\odot\mathbf{B}\right)\odot^{\mathrm{T}}\left(\mathbf{C}\odot\mathbf{D}\right) = \left(\mathbf{A}\odot^{\mathrm{T}}\mathbf{C}\right)\odot\left(\mathbf{B}\odot^{\mathrm{T}}\mathbf{D}\right), \tag{A.3}$$

$$\operatorname{vec}\left(\mathbf{ABC}\right) = \left(\mathbf{C}^{\mathrm{T}}\otimes\mathbf{A}\right)\operatorname{vec}\left(\mathbf{B}\right), \tag{A.4}$$

$$\mathbf{AB}\odot^{\mathrm{T}}\mathbf{CD} = \left(\mathbf{A}\odot^{\mathrm{T}}\mathbf{C}\right)\otimes\left(\mathbf{B}\otimes\mathbf{D}\right). \tag{A.5}$$

We define a permutation matrix $\mathbf{P}^{(n)}$ that permutes the nth mode of a tensor to the first mode. It holds that ${\mathbf{P}^{(n)}}^{\mathrm{T}}\mathbf{P}^{(n)} = \mathbf{I}$ [124]. We use the following identity:

$$\begin{aligned}\mathbf{P}^{(n)}\operatorname{vec}\left(\left\llbracket\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(N)}\right\rrbracket\right) &= \\ \operatorname{vec}\left(\left\llbracket\mathbf{U}^{(n)},\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(n-1)},\mathbf{U}^{(n+1)},\ldots,\mathbf{U}^{(N)}\right\rrbracket\right)&.\end{aligned} \tag{A.6}$$

Denoting $\mathbf{V}^{\{n\}} = \odot_{q=1,q\neq N-n+1}^{N}\mathbf{U}^{(N-q+1)}$, we can also define the following two identities:

$$\begin{aligned}{\mathbf{P}^{(n)}}^{\mathrm{T}}\left(\mathbf{V}^{\{n\}}\otimes\mathbf{I}\right)\operatorname{vec}\left(\mathbf{X}\right) &= \\ \operatorname{vec}\left(\left\llbracket\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(n-1)},\mathbf{X},\mathbf{U}^{(n+1)},\ldots,\mathbf{U}^{(N)}\right\rrbracket\right)&,\end{aligned} \tag{A.7}$$

$${\mathbf{P}^{(n)}}^{\mathrm{T}}\operatorname{vec}\left(\mathbf{U}^{(n)}{\mathbf{V}^{\{n\}}}^{\mathrm{T}}\right) = \operatorname{vec}\left(\left\llbracket\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(n)}\right\rrbracket\right). \tag{A.8}$$

Finally, we also use the following derivative [124]:

$$\frac{\partial\operatorname{vec}\left(\left\llbracket\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(n)}\right\rrbracket\right)}{\partial\operatorname{vec}\left(\mathbf{U}^{(n)}\right)} = {\mathbf{P}^{(n)}}^{\mathrm{T}}\left(\mathbf{V}^{\{n\}}\otimes\mathbf{I}_{I_n}\right). \tag{A.9}$$

A.2 WLS for CPD using low-rank weights

The computation of a rank-R CPD $\left\llbracket\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(N)}\right\rrbracket$ of a tensor $\mathcal{T}$ using a weighted least-squares approach with known weight tensor $\mathcal{W}$, leads to the following optimization problem:

$$\min_{\mathbf{z}} f = \frac{1}{2}\left|\left|\mathcal{F}\right|\right|_{\mathrm{F}}^{2} \quad \text{with} \quad \mathcal{F} = \mathcal{W} * \left(\left\llbracket\mathbf{U}^{(1)},\ldots,\mathbf{U}^{(N)}\right\rrbracket - \mathcal{T}\right), \tag{A.10}$$

in which the variables $\mathbf{U}^{(n)}$, for $1 \leq n \leq N$, are concatenated in a vector $\mathbf{z} = \left[\operatorname{vec}\left(\mathbf{U}^{(1)}\right); \cdots; \operatorname{vec}\left(\mathbf{U}^{(N)}\right)\right] \in \mathbb{K}^{RI^{+}}$ with $I^{+} = \sum_{n=1}^{N} I_n$. Assume $\mathcal{W}$

admits a rank-L PD:

$$\mathcal{W} = \left[\!\left[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}\right]\!\right] \quad \text{with} \quad \mathbf{A}^{(n)} \in \mathbb{K}^{I_n \times L}. \tag{A.11}$$

Note that (A.10)-(A.11) reduces to regular least squares when $L = 1$ and $\mathbf{a}^{(1)} = \mathbf{1}_{I_n}$ for $1 \leq n \leq N$. As such, the rank-L PD structure enables more general weighting schemes.

In order to solve (A.10)-(A.11), one can use standard qN and NLS algorithms, which require expressions for the evaluation of the objective function, gradient, Jacobian, Jacobian-vector product, Gramian, and Gramian-vector product. In this section, we derive expressions for these ingredients which explicitly exploit the low-rank structure of the weight tensor. Although we focus on the implementation of the Gauss–Newton (GN) method, which is a particular type of NLS algorithm [150], the expressions can be used for other qN and NLS algorithms as well. We use the complex optimization framework from [177], [179] to implement the GN method. The framework provides qN and NLS implementations as well as line search, plane search, and trust-region methods. Additionally, we give an overview of the per-iteration complexity of each ingredient.

The GN method using dogleg trust-region solves (A.10)-(A.11) by linearizing the residual $\text{vec}\left(\mathcal{F}\right)$ in each iteration k and subsequently solving the following least-squares problem [150]:

$$\min_{\mathbf{p}_k} \frac{1}{2} \left|\left|\text{vec}\left(\mathcal{F}_k\right) + \mathbf{J}_k \mathbf{p}_k\right|\right|_{\text{F}}^2 \quad \text{s.t.} \quad \left|\left|\mathbf{p}_k\right|\right| \leq \Delta_k \tag{A.12}$$

with step $\mathbf{p}_k = \mathbf{z}_{k+1} - \mathbf{z}_k$, Jacobian $\mathbf{J} = \text{d}\,\text{vec}\left(\mathcal{F}\right)/\text{d}\mathbf{z}$, and trust-region Δ_k. The exact solution to (A.12) is given by the linear system $\mathbf{H}_k \mathbf{p}_k = -\overline{\mathbf{g}}_k$ with $\mathbf{H}$ the Hessian, which we approximate with the Gramian of the Jacobian, and the conjugated gradient $\overline{\mathbf{g}} = \left(\partial f / \partial \mathbf{z}\right)^{\text{H}}$ [150]. The variables can then be updated as $\mathbf{z}_{k+1} = \mathbf{z}_k + \mathbf{p}_k$. In this appendix, we solve the linear system using several preconditioned conjugate gradient (PCG) iterations in order to reduce computational complexity. The GN method is summarized in Algorithm A.1.

A.2.1 Objective function

The objective function f can be evaluated as follows:

$$f = \frac{1}{2} \left|\left| \left[\!\left[\tilde{\mathbf{U}}^{(1)}, \tilde{\mathbf{U}}^{(2)}, \ldots, \tilde{\mathbf{U}}^{(N)}\right]\!\right] - \tilde{\mathcal{T}} \right|\right|_{\text{F}}^2, \tag{A.13}$$

using the weighted factor matrices $\tilde{\mathbf{U}}^{(n)} \in \mathbb{K}^{I_n \times RL}$, which are defined by $\tilde{\mathbf{U}}^{(n)} = \mathbf{A}^{(n)} \odot^{\text{T}} \mathbf{U}^{(n)}$, for $1 \leq n \leq N$. Importantly, the weighted factor matrices are computed only once per iteration. The weighted tensor $\tilde{\mathcal{T}} = \mathcal{W} * \mathcal{T}$ can be computed beforehand.

Algorithm A.1: Weighted Least Squares with CPD-constrained weight tensor using Gauss–Newton and dogleg trust region

```
Input: $\mathcal{T}$, $\{\mathbf{A}^{(n)}\}_{n=1}^{N}$, and initial estimate for $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$
Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$
while not converged do
    Compute gradient $\mathbf{g}$ using (A.15) and (A.16).
    Use PCG to solve $\mathbf{Hp} = -\overline{\mathbf{g}}$ for $\mathbf{p}$ using Gramian-vector products in (A.18) and a block-Jacobi preconditioner as explained in Appendix A.2.5.
    Update $\mathbf{U}^{(n)}$, for $1 \leq n \leq N$, using dogleg trust region from $\mathbf{p}$, $\mathbf{g}$, and function evaluation (A.13).
end while
```

A.2.2 Jacobian

The Jacobian $\mathbf{J}$ can be partitioned in the following way:

$$\mathbf{J} = \frac{\mathrm{d}\,\mathrm{vec}\left(\mathcal{F}\right)}{\mathrm{d}\mathbf{z}} = \begin{bmatrix}\mathbf{J}^{(1)} & \mathbf{J}^{(2)} & \cdots & \mathbf{J}^{(N)}\end{bmatrix} \in \mathbb{K}^{I^{\times} \times RI^{+}}$$

with $I^{\times} = \prod_{n=1}^{N} I_n$. The nth sub-Jacobian $\mathbf{J}^{(n)}$ is defined by:

$$\mathbf{J}^{(n)} = \mathbf{P}^{(n)^{\mathrm{T}}} \left(\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \mathbf{M}^{(n)} \in \mathbb{K}^{I^{\times} \times RI_n},$$

using $\tilde{\mathbf{V}}^{\{n\}} = \odot_{q=1,q\neq N-n+1}^{N} \tilde{\mathbf{U}}^{(N-q+1)}$ and matrix $\mathbf{M}^{(n)}$:

$$\mathbf{M}^{(n)} = \begin{bmatrix} \mathbf{I}_R \otimes \mathrm{diag}\left(\mathbf{a}_1^{(n)}\right) \\ \mathbf{I}_R \otimes \mathrm{diag}\left(\mathbf{a}_2^{(n)}\right) \\ \vdots \\ \mathbf{I}_R \otimes \mathrm{diag}\left(\mathbf{a}_L^{(n)}\right) \end{bmatrix} \in \mathbb{K}^{RLI_n \times RI_n}. \tag{A.14}$$

Proof. First, define matrix $\mathbf{D} = \mathrm{diag}\left(\mathrm{vec}\left(\mathcal{W}\right)\right) \in \mathbb{K}^{I^{\times} \times I^{\times}}$. Using (A.9), one can show that nth sub-Jacobian equals:

$$\begin{aligned}\mathbf{J}^{(n)} = \frac{\partial \mathrm{vec}\left(\mathcal{F}\right)}{\partial \mathrm{vec}\left(\mathbf{U}^{(n)}\right)} &= \mathbf{D}\frac{\partial \mathrm{vec}\left(\left[\!\left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}\right]\!\right]\right)}{\partial \mathrm{vec}\left(\mathbf{U}^{(n)}\right)} \\ &= \mathbf{D}\mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right).\end{aligned}$$

Define $\mathbf{B}^{\{n\}} = \odot_{q=1,q\neq N-n+1}^{N} \mathbf{A}^{(N-q+1)}$. The CPD structure of the weight tensor $\mathcal{W}$ can then be exploited as follows:

$$\begin{aligned}
\mathbf{J}^{(n)} &= \operatorname{diag}\left(\operatorname{vec}\left(\left[\!\left[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}\right]\!\right]\right)\right) \mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \\
&= \operatorname{diag}\left(\mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{B}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \operatorname{vec}\left(\mathbf{A}^{(n)}\right)\right) \mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \\
&= \left[\mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{B}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \operatorname{vec}\left(\mathbf{A}^{(n)}\right)\right] \odot^{\mathrm{T}} \left[\mathbf{P}^{(n)^{\mathrm{T}}} \left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)\right] \\
&= \mathbf{P}^{(n)^{\mathrm{T}}} \left(\left(\mathbf{B}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \odot^{\mathrm{T}} \left(\mathbf{V}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)\right) \left[\operatorname{vec}\left(\mathbf{A}^{(n)}\right) \otimes \mathbf{I}_{RI_n}\right] \\
&= \mathbf{P}^{(n)^{\mathrm{T}}} \left(\left(\mathbf{B}^{\{n\}} \odot^{\mathrm{T}} \mathbf{V}^{\{n\}}\right) \otimes \mathbf{I}_{I_n}\right) \mathbf{M}^{(n)} \\
&= \mathbf{P}^{(n)^{\mathrm{T}}} \left(\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right) \mathbf{M}^{(n)}.
\end{aligned}$$

Identities (A.8) and (A.5) are used to obtain the second and fourth equation. One can obtain the second to last equation using (A.3) and (A.14). Note that we do not explicitly compute the Jacobian.

A.2.3 Gradient

The gradient $\mathbf{g}$ can be partitioned in the following way:

$$\mathbf{g} = \begin{bmatrix} \mathbf{g}^{(1)} & \mathbf{g}^{(2)} & \cdots & \mathbf{g}^{(N)} \end{bmatrix} \in \mathbb{K}^{RI^{+}}, \tag{A.15}$$

in which $\mathbf{g}^{(n)} \in \mathbb{K}^{RI_n}$ is defined by:

$$\mathbf{g}^{(n)} = \mathbf{M}^{(n)^{\mathrm{T}}} \operatorname{vec}\left(\tilde{\mathbf{U}}^{(n)} \tilde{\mathbf{V}}^{\{n\}^{\mathrm{T}}} \tilde{\mathbf{V}}^{\{n\}} - \tilde{\mathbf{T}}_{(n)} \tilde{\mathbf{V}}^{\{n\}}\right). \tag{A.16}$$

Proof. The nth sub-gradient is given by:

$$\begin{aligned}
\mathbf{g}^{(n)} &= \frac{\partial f}{\partial \operatorname{vec}\left(\mathbf{U}^{(n)}\right)} = \mathbf{J}^{(n)^{\mathrm{T}}} \overline{\operatorname{vec}\left(\mathcal{F}\right)} \\
&= \mathbf{M}^{(n)^{\mathrm{T}}} \left(\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)^{\mathrm{T}} \mathbf{P}^{(n)} \cdot \mathbf{P}^{(n)^{\mathrm{T}}} \overline{\operatorname{vec}\left(\tilde{\mathbf{U}}^{(n)} \tilde{\mathbf{V}}^{\{n\}^{\mathrm{T}}} - \tilde{\mathbf{T}}_{(n)}\right)} \\
&= \mathbf{M}^{(n)^{\mathrm{T}}} \left[\left(\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)^{\mathrm{T}} \operatorname{vec}\left(\overline{\tilde{\mathbf{U}}^{(n)}} \tilde{\mathbf{V}}^{\{n\}^{\mathrm{H}}}\right) - \left(\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)^{\mathrm{T}} \overline{\tilde{\mathbf{T}}_{(n)}}\right] \\
&= \mathbf{M}^{(n)^{\mathrm{T}}} \operatorname{vec}\left(\overline{\tilde{\mathbf{U}}^{(n)}} \tilde{\mathbf{V}}^{\{n\}^{\mathrm{H}}} \tilde{\mathbf{V}}^{\{n\}} - \overline{\tilde{\mathbf{T}}_{(n)}} \tilde{\mathbf{V}}^{\{n\}}\right).
\end{aligned}$$

using identities (A.4) and (A.7) to obtain the last equation. The matrices $\mathbf{M}^{(n)}$, $1 \leq n \leq N$, only depend on a factor matrix of the weight tensor and can therefore be computed beforehand.

A.2.4 Gramian-vector product

First, we derive an efficient way to compute the product of a sub-Jacobian $\mathbf{J}^{(n)}$ and $\operatorname{vec}\left(\mathbf{X}^{(n)}\right)$ with $\mathbf{X}^{(n)} \in \mathbb{K}^{I_n \times R}$:

$$\begin{aligned}
\mathbf{J}^{(n)}\operatorname{vec}\left(\mathbf{X}^{(n)}\right) &= \mathbf{P}^{(n)^{\mathrm{T}}}\left(\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)\mathbf{M}^{(n)}\operatorname{vec}\left(\mathbf{X}^{(n)}\right) \\
&= \mathbf{P}^{(n)^{\mathrm{T}}}\left(\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)\operatorname{vec}\left(\tilde{\mathbf{X}}^{(n)}\right) \\
&= \operatorname{vec}\left(\left\llbracket \tilde{\mathbf{U}}^{(1)}, \ldots, \tilde{\mathbf{U}}^{(n-1)}, \tilde{\mathbf{X}}^{(n)}, \tilde{\mathbf{U}}^{(n+1)}, \ldots, \tilde{\mathbf{U}}^{(N)} \right\rrbracket\right)
\end{aligned} \tag{A.17}$$

with $\tilde{\mathbf{X}}^{(n)} = \mathbf{A}^{(n)} \odot^{\mathrm{T}} \mathbf{X}^{(n)}$ using identity (A.8) to obtain the last equation. Next, we derive an efficient expression for the Gramian-vector product:

$$\begin{aligned}
\mathbf{r} &= \mathbf{J}^{(m)^{\mathrm{H}}}\mathbf{J}^{(n)}\operatorname{vec}\left(\mathbf{X}^{(n)}\right) \\
&= \mathbf{M}^{(m)^{\mathrm{H}}}\left(\tilde{\mathbf{V}}^{\{m\}} \otimes \mathbf{I}_{I_m}\right)^{\mathrm{H}} \\
&\qquad \cdot \mathbf{P}^{(m)}\operatorname{vec}\left(\left\llbracket \tilde{\mathbf{U}}^{(1)}, \ldots, \tilde{\mathbf{U}}^{(n-1)}, \tilde{\mathbf{X}}^{(n)}, \tilde{\mathbf{U}}^{(n+1)}, \ldots, \tilde{\mathbf{U}}^{(N)} \right\rrbracket\right) \\
&= \mathbf{M}^{(m)^{\mathrm{H}}}\left(\tilde{\mathbf{V}}^{\{m\}} \otimes \mathbf{I}_{I_m}\right)^{\mathrm{H}}\operatorname{vec}\left(\left\llbracket \tilde{\mathbf{U}}^{(m)}, \tilde{\mathbf{U}}^{(1)}, \ldots, \tilde{\mathbf{X}}^{(n)}, \ldots, \tilde{\mathbf{U}}^{(N)} \right\rrbracket\right) \\
&= \mathbf{M}^{(m)^{\mathrm{H}}}\left(\tilde{\mathbf{V}}^{\{m\}} \otimes \mathbf{I}_{I_m}\right)^{\mathrm{H}}\left(\hat{\mathbf{V}}^{\{m\}} \otimes \mathbf{I}_{I_m}\right)\operatorname{vec}\left(\tilde{\mathbf{U}}^{(m)}\right) \\
&= \mathbf{M}^{(m)^{\mathrm{H}}}\left(\tilde{\mathbf{V}}^{\{m\}^{\mathrm{H}}}\hat{\mathbf{V}}^{\{m\}} \otimes \mathbf{I}_{I_m}\right)\operatorname{vec}\left(\tilde{\mathbf{U}}^{(m)}\right)
\end{aligned} \tag{A.18}$$

with $\hat{\mathbf{V}}^{\{m\}} = \tilde{\mathbf{U}}^{(N)} \odot \cdots \odot \tilde{\mathbf{U}}^{(n+1)} \odot \tilde{\mathbf{X}}^{(n)} \odot \tilde{\mathbf{U}}^{(n-1)} \odot \cdots \odot \tilde{\mathbf{U}}^{(1)}$ in which m indicates that the mth factor matrix is omitted in the Khatri–Rao product. We used (A.17), (A.6), (A.7) and (A.2) in the subsequent steps of (A.18). If $m = n$, one can show that (A.18), using $\mathbf{Q}^{\{n\}} = \tilde{\mathbf{V}}^{\{n\}^{\mathrm{T}}}\tilde{\mathbf{V}}^{\{n\}} \in \mathbb{K}^{RL \times RL}$, reduces to:

$$\mathbf{J}^{(n)^{\mathrm{H}}}\mathbf{J}^{(n)}\operatorname{vec}\left(\mathbf{X}^{(n)}\right) = \mathbf{M}^{(n)^{\mathrm{H}}}\operatorname{vec}\left(\tilde{\mathbf{X}}^{(n)}\mathbf{Q}^{\{n\}^{\mathrm{T}}}\right).$$

A.2.5 Block-Jacobi preconditioner

We use a block-Jacobi preconditioner to reduce the number of conjugate gradient (CG) iterations and improve overall convergence. In that case, one has to compute the inverse of $\mathbf{J}^{(n)^{\mathrm{H}}}\mathbf{J}^{(n)} \in \mathbb{K}^{RI_n \times RI_n}$, for $1 \leq n \leq N$, in each

Table A.1: *Per-iteration computational complexity of the ingredients of the NLS algorithm.*

	Calls/iteration	Complexity
Weighted factor matrices	1	$\mathcal{O}(NRLI)$
Objective function	$1 + \mathrm{it}_{\mathrm{TR}}$	$\mathcal{O}(RLI^N)$
Gradient	1	$\mathcal{O}(NRLI^N + NR^2LI^2)$
Gramian-vector	$\mathrm{it}_{\mathrm{CG}}$	$\mathcal{O}(NR^2L^2I + NR^2LI^2)$

iteration. First, we derive an expression for the (n,n)th sub-Gramian:

$$\begin{aligned}\mathbf{J}^{(n)^{\mathrm{H}}}\mathbf{J}^{(n)} &= \mathbf{M}^{(n)^{\mathrm{H}}}\left(\tilde{\mathbf{V}}^{(n)} \otimes \mathbf{I}_{I_n}\right)^{\mathrm{H}}\left(\tilde{\mathbf{V}}^{(n)} \otimes \mathbf{I}_{I_n}\right)\mathbf{M}^{(n)} \\ &= \mathbf{M}^{(n)^{\mathrm{H}}}\left(\tilde{\mathbf{V}}^{\{n\}^{\mathrm{T}}}\tilde{\mathbf{V}}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)\mathbf{M}^{(n)} \\ &= \mathbf{M}^{(n)^{\mathrm{H}}}\left(\mathbf{Q}^{\{n\}} \otimes \mathbf{I}_{I_n}\right)\mathbf{M}^{(n)}.\end{aligned}$$

with $\mathbf{Q}^{\{n\}} = \tilde{\mathbf{V}}^{\{n\}^{\mathrm{T}}}\tilde{\mathbf{V}}^{\{n\}} \in \mathbb{K}^{RL\times RL}$. Using identity (A.1), we obtain $\mathbf{Q}^{\{n\}} = *_{q=1,q\neq n}^{N} \tilde{\mathbf{U}}^{(N-q+1)^{\mathrm{H}}}\tilde{\mathbf{U}}^{(N-q+1)} \in \mathbb{K}^{RL\times RL}$. The inverse of $\mathbf{J}^{(n)^{\mathrm{H}}}\mathbf{J}^{(n)}$ can then be computed efficiently as:

$$\left(\mathbf{J}^{(n)^{\mathrm{H}}}\mathbf{J}^{(n)}\right)^{\dagger} = \left(\mathbf{M}^{(n)}\right)^{\dagger}\left(\left(\mathbf{Q}^{\{n\}}\right)^{\dagger} \otimes \mathbf{I}_{I_n}\right)\left(\mathbf{M}^{(n)}\right)^{\dagger,\mathrm{H}},$$

in which $\left(\mathbf{M}^{(n)}\right)^{\dagger}$ can be computed beforehand and $\left(\mathbf{Q}^{\{n\}}\right)^{\dagger}$ requires the inverse of small matrices of size $(RL \times RL)$.

A.2.6 Complexity

In Table A.1, we report the per-iteration complexity for the ingredients of Algorithm A.1. We assume that $I_n = I$ for $1 \leq n \leq N$. The complexity is similar to regular LS algorithms. In comparison to [180], the factor R is replaced by RL and there is an additional cost for the multiplication with $\mathbf{M}^{(n)}$.

A.3 Direction-of-arrival estimation using WLS

Direction-of-arrival (DOA) estimation is an important problem in radar, sonar, and telecommunication applications [127]. We show that the performance can be improved by including prior knowledge about the accuracy of the sensors via WLS.

When considering a uniform linear array (ULA) with line-of-sight signals impinging from the far field, the DOA estimation problem can be reformulated as the computation of a CPD [141], [160], [165], [167]. Consider a

Table A.2: *Incorporating prior knowledge about the accuracy of the sensors in DOA estimation using weighted least squares allows one to decrease the relative error on the DOAs.*

	Least squares	Weighted least squares	
		rank-1	rank-2
Relative error	0.0322	0.0187	**0.0122**

ULA with M uniformly spaced and omnidirectional sensors receiving length-K signals from R narrow-band sources in the far field. In that case, the problem can be described by $\mathbf{X} = \mathbf{MS}$ in which $\mathbf{X} \in \mathbb{K}^{M\times K}$ contains the observed measurements, $\mathbf{M} \in \mathbb{K}^{M\times R}$ contains the so-called steering vectors, and $\mathbf{S} \in \mathbb{K}^{R\times K}$ contains the sources. The steering vectors are defined element-wise as $m_{mr} = \theta_r^{m-1}$ with $\theta_r = e^{-2\pi i\Delta\sin(\alpha_r)\lambda^{-1}}$. The inter-element spacing is denoted by Δ, the rth DOA with respect to the normal is denoted by α_r (i.e., -90° $\leq \alpha_r \leq$ 90°), and the wavelength is denoted by λ. One way to obtain a tensor is to reshape $\mathbf{X}$ into a third-order tensor $\mathcal{X} \in \mathbb{K}^{I\times J\times K}$ such that $M = IJ$ using segmentation; see [21], [62]. The obtained tensor $\mathcal{X}$ admits a CPD $[\![\mathbf{A}, \mathbf{B}, \mathbf{S}]\!]$ in which $\mathbf{A}$ and $\mathbf{B}$ are Vandermonde matrices from which the DOAs can be extracted.

Consider a ULA with $M = 25$ sensors and $R = 2$ source signals of length $K = 100$. Assume there are two types of sensors with different quality: the SNR on the first seventeen sensors is 25 dB and the SNR on the last eight sensors is 5 dB. In order to accommodate for the difference in SNR between the sensors, one could use the following weight tensor $\mathcal{W} = \text{unvec}\,(\mathbf{w})$ with $\mathbf{w}_{(k-1)M+1:(k-1)M+17} = \mathbf{1}_{17}$ and $\mathbf{w}_{(k-1)M+18:(k-1)M+25} = 0.01 \cdot \mathbf{1}_8$ for $1 \leq k \leq K$. We model the weight tensor with a rank-L CPD using $L = \{1, 2\}$; note that the weight tensor has rank two. The relative error $\epsilon_{\boldsymbol{\alpha}}$ of the DOAs is defined by $||\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}||_{\text{F}} \,/\, ||\boldsymbol{\alpha}||_{\text{F}}$ with $\boldsymbol{\alpha}$ a vector containing the R DOAs. In Table A.2 we report the median relative error across 50 experiments. It is clear that WLS outperforms LS, even when using a rank-1 model for the weight tensor. Also, using an additional rank-1 term to model the weight tensor further decreases the relative error.

A.4 Conclusion

A new WLS-based approach has been proposed for the computation of a CPD using low-rank weights. In particular, the weight tensor was modeled as a PD, enabling efficient weighting schemes. We have derived expressions for the ingredients of well-known qN and NLS algorithms that carefully exploit the CPD structure of the weight tensor. Also, a complexity analysis was performed for the NLS-type algorithms. We validated our algorithm for DOA estimation in ULAs, outperforming regular least squares. In future work, one can derive WLS-based algorithms for other tensor decompositions. Additionally, other tensor models for the weight tensor can be considered such as the multilinear singular value decomposition, tensor train, or hierachical Tucker model [54], [95], [152].

Bibliography

[1] K. Abed–Meraim, W. Qiu, and Y. Hua, "Blind system identification", *Proceedings of the IEEE*, vol. 85, no. 8, pp. 1310–1322, Aug. 1997.

[2] E. Acar, C. A. Bingol, H. Bingol, R. Bro, and B. Yener, "Multiway analysis of epilepsy tensors", *Bioinformatics*, vol. 23, no. 13, pp. i10–i18, Jul. 2007. eprint: `http://bioinformatics.oxfordjournals.org/content/23/13/i10.full.pdf+html`.

[3] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations with missing data", in *SIAM International Conference on Data Mining*, 2010, pp. 701–712.

[4] E. Acar, D. Dunlavy, and T. Kolda, "A scalable optimization approach for fitting canonical tensor decompositions", *Journal of Chemometrics*, vol. 25, no. 2, pp. 67–86, 2011.

[5] E. Acar and B. Yener, "Unsupervised multiway data analysis: A literature survey", *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 1, pp. 6–20, Jan. 2009.

[6] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models", *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2773–2832, 2014.

[7] E.-W. Bai and Y. Liu, "On the least squares solutions of a system of bilinear equations", in *Proceedings of the 44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference (CDC-ECC 2005, Seville, Spain)*, Dec. 2005, pp. 1197–1202.

[8] J. Ballani and L. Grasedyck, "A projection method to solve linear systems in tensor format", *Numerical Linear Algebra with Applications*, vol. 20, pp. 27–43, Jan. 2013.

[9] K. Batselier and N. Wong, "A constructive arbitrary-degree Kronecker product decomposition of tensors", *Numerical Linear Algebra with Applications*, vol. 24, no. 5, e2097, Oct. 2017.

[10] H. Becker, L. Albera, P. Comon, M. Haardt, and G. Birot, "EEG extended source localization: Tensor-based vs. conventional methods", *NeuroImage*, vol. 96, pp. 143–157, 2014.

[11] A. Belouchrani, K. Abed–Meraim, J.-F. Cardoso, and E. Moulines, "A blind source separation technique using second-order statistics", *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 434–444, Feb. 1997.

[12] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks", *Science*, vol. 353, no. 6298, pp. 163–166, 2016.

[13] A. Bertrand, "Distributed signal processing for wireless EEG sensor networks", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 6, pp. 923–935, Dec. 2015.

[14] G. Beylkin and M. J. Mohlenkamp, “Algorithms for numerical analysis in high dimensions”, *SIAM Journal on Scientific Computing*, vol. 26, no. 6, pp. 2133–2159, 2005.

[15] H. N. Bharath, D. M. Sima, N. Sauwen, U. Himmelreich, L. De Lathauwer, and S. Van Huffel, “Nonnegative canonical polyadic decomposition for tissue-type differentiation in gliomas”, *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 4, pp. 1124–1132, Jul. 2017.

[16] M. Boussé and L. De Lathauwer, “Nonlinear least squares algorithm for canonical polyadic decomposition using low-rank weights”, in *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2017, Curaçao, Dutch Antilles)*, Dec. 2017, pp. 39–43.

[17] ——, “Large-scale autoregressive system identification using Kronecker product equations”, in *2018 6th IEEE Global Conference on Signal and Information Processing (GlobalSIP 2018, Anaheim, California, USA)*, Nov. 2018, pp. 1348–1352.

[18] M. Boussé, O. Debals, and L. De Lathauwer, “A novel deterministic method for large-scale blind source separation”, in *Proceedings of the 23rd European Signal Processing Conference (EUSIPCO 2015, Nice, France)*, Aug. 2015, pp. 1935–1939.

[19] ——, “A tensor-based method for large-scale blind system identification using segmentation”, in *Proceedings of the 24th European Signal Processing Conference (EUSIPCO 2016, Budapest, Hungary)*, Sep. 2016, pp. 2015–2019.

[20] ——, “Tensor-based large-scale blind system identification using segmentation”, *IEEE Transactions on Signal Processing*, vol. 65, no. 21, pp. 5770–5784, Nov. 2017.

[21] ——, “A tensor-based method for large-scale blind source separation using segmentation”, *IEEE Transactions on Signal Processing*, vol. 65, no. 2, pp. 346–358, Jan. 2017.

[22] M. Boussé, I. Domanov, and L. De Lathauwer, “Linear systems with a multilinear singular value decomposition constrained solution”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 17-56, 2017.

[23] M. Boussé, G. Goovaerts, N. Vervliet, O. Debals, S. Van Huffel, and L. De Lathauwer, “Irregular heartbeat classification using Kronecker product equations”, in *39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2017, Jeju Island, South-Korea)*, Jul. 2017, pp. 438–441.

[24] M. Boussé, N. Sidiropoulos, and L. De Lathauwer, “NLS algorithm for Kronecker-structured linear systems with a CPD constrained solution”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-13, 2019, Accepted for publication in the proceedings of the 27th European Signal Processing Conference (EUSIPCO 2019, A Coruña, Spain).

[25] M. Boussé, N. Vervliet, O. Debals, and L. De Lathauwer, “Face recognition as a Kronecker product equation”, in *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2017, Curaçao, Dutch Antilles)*, Dec. 2017, pp. 276–280.

[26] M. Boussé, N. Vervliet, I. Domanov, O. Debals, and L. De Lathauwer, “Linear systems with a canonical polyadic decomposition constrained solution: Algorithms and applications”, *Numerical Linear Algebra with Applications*, vol. 25, no. 6, e2190, Aug. 2018.

[27] M. Brazell, N. Li, C. Navasca, and C. Tamon, “Solving multilinear systems via tensor inversion”, *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 2, pp. 542–570, May 2013.

[28] R. Bro and C. Andersson, “Improving the speed of multiway algorithms: Part II: Compression”, *Chemometrics and Intelligent Laboratory Systems*, vol. 42, no. 12, pp. 105–113, 1998.

[29] A. Browet and P. Van Dooren, “Low-rank similarity measure for role model extraction”, in *21st International Symposium on Mathematical Theory of Networks and Systems (MNTS 2017, Groningen, The Netherlands)*, Jul. 2014, pp. 1412–1418.

[30] C. F. Caiafa and A. Cichocki, “Generalizing the column-row matrix decomposition to multi-way arrays”, *Linear Algebra and its Applications*, vol. 433, no. 3, pp. 557–573, Sep. 2010.

[31] ——, “Multidimensional compressed sensing and their applications”, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 3, no. 6, pp. 355–380, Nov. 2013.

[32] D. Callaerts, “Signal separation methods based on singular value decomposition and their application to the real-time extraction of the fetal electrocardiogram from cutaneous recordings”, PhD thesis, KU Leuven, Dec. 1989.

[33] E. J. Candés and Y. Plan, “Matrix completion with noise”, *Proceedings of the IEEE*, vol. 98, no. 6, pp. 925–936, Jun. 2010.

[34] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling”, *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[35] J. D. Carrol, S. Pruzansky, and J. B. Kruskal, “CANDELINC: A general approach to multidimensional analysis of many-way arrays with linear constraints on parameters”, *Psychometrika*, vol. 45, no. 1, pp. 3–24, Mar. 1980.

[36] C. Chatzichristos, E. Kofidis, and S. Theodoridis, “PARAFAC2 and its block term decomposition analog for blind fMRI source unmixing”, in *Proceedings of the 25th European Signal Processing Conference (EUSIPCO 2017, Kos island, Greece)*, Aug. 2017, pp. 2081–2085.

[37] P. T. Chavda and S. Solanki, “Illumination invariant face recognition based on PCA (eigenface)”, *International Journal of Engineering Development and Research*, vol. 2, no. 2, pp. 2155–2162, Jun. 2014.

[38] L. Chiantini and G. Ottaviani, “On generic identifiability of 3-tensors of small rank”, *SIAM Journal on Matrix Analysis and Applications*, vol. 33, no. 3, pp. 1018–1037, 2012.

[39] L. Chiantini, G. Ottaviani, and N. Vannieuwenhoven, “An algorithm for generic and low-rank specific identifiability of complex tensors”, *SIAM Journal on Matrix Analysis and Applications*, vol. 35, no. 4, pp. 1265–1287, 2014.

[40] A. Cichocki and S. Amari, *Adaptive blind signal and image processing: Learning algorithms and applications.* John Wiley Chichester, 2002, vol. 1.

[41] A. Cichocki, D. P. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. F. Caiafa, and A.-H. Phan, “Tensor decompositions for signal processing applications: From two-way to multiway component analysis”, *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, Mar. 2015.

[42] A. Cichocki, R. Zdunek, A. Phan, and S. Amari, *Nonnegative matrix and tensor factorizations: Applications to exploratory multi-way data analysis and blind source separation.* Wiley, 2009.

[43] G. D. Clifford, F. Azuaje, and P. McSharry, *Advanced Methods And Tools for ECG Data Analysis.* Norwood, MA, USA: Artech House, Inc., 2006.

[44] S. Cohen and C. Tomasi, “Systems of bilinear equations”, Technical Report, 1997.

[45] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent Component Analysis and Applications.* Academic press, 2010.

[46] R. Costantini, L. Sbaiz, and S. Susstrunk, “Higher order SVD analysis for dynamic texture synthesis”, *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 42–52, Jan. 2008.

[47] L. De Lathauwer, “A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization”, *SIAM Journal on Matrix Analysis and Applications*, vol. 28, no. 3, pp. 642–666, Sep. 2006.

[48] ——, “Decompositions of a higher-order tensor in block terms — Part I: Lemmas for partitioned matrices”, *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1022–1032, Sep. 2008.

[49] ——, “Decompositions of a higher-order tensor in block terms — Part II: Definitions and uniqueness”, *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1033–1066, Sep. 2008.

[50] ——, “A short introduction to tensor-based methods for factor-analysis and blind source separation”, *Proceedings of the 7th International Symposium on Image and Signal Processing and Analysis (ISPA 2011, Dubrovnik, Croatia)*, pp. 4–6, Sep. 2011.

[51] ——, “Blind separation of exponential polynomials and the decomposition of a tensor in rank-$(L_r, L_r, 1)$ terms”, *SIAM Journal on Matrix Analysis and Applications*, vol. 32, no. 4, pp. 1451–1474, Dec. 2011.

[52] ——, “Block component analysis, a new concept for blind source separation”, in *Latent Variable Analysis and Signal Separation*, ser. Lecture Notes in Computer Science, vol. 7191, Springer Berlin/Heidelberg, 2012, pp. 1–8.

[53] L. De Lathauwer and A. De Baynast, “Blind deconvolution of DS-CDMA signals by means of decomposition in rank-$(1, L, L)$ terms”, *IEEE Transactions on Signal Processing*, vol. 56, no. 4, pp. 1562–1571, Apr. 2008.

[54] L. De Lathauwer, B. De Moor, and J. Vandewalle, “A multilinear singular value decomposition”, *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, Jul. 2000.

[55] ——, “Fetal electrocardiogram extraction by blind source subspace separation”, *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 5, pp. 567–572, May 2000.

[56] ——, “On the best rank-1 and rank-$R_1, R_2, \cdots, R_N$ approximation of higher-order tensors”, *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, Jul. 2000.

[57] L. De Lathauwer, B. D. Moor, and J. Vandewalle, “An introduction to independent component analysis”, *Journal of chemometrics*, vol. 14, pp. 123–149, May 2000.

[58] L. De Lathauwer and D. Nion, “Decompositions of a higher-order tensor in block terms — Part III: Alternating least squares algorithms”, *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1067–1083, Sep. 2008.

[59] 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 and its Applications*, vol. 391, pp. 31–55, Nov. 2004.

[60] B. De Moor, “The singular value decompositions and long and short spaces of noisy matrices”, *IEEE Transactions on Signal Processing*, vol. 41, no. 9, pp. 2826–2838, Sep. 1993.

[61] M. De Vos, A. Vergult, L. De Lathauwer, W. De Clerq, S. Van Huffel, P. Dupont, A. Palmini, and W. Van Paesschen, “Canonical decomposition of ictal scalp EEG reliably detects the seizure onset zone”, *NeuroImage*, vol. 37, no. 3, pp. 844–854, May 2007.

[62] O. Debals and L. De Lathauwer, “Stochastic and deterministic tensorization for blind signal separation”, in *Latent Variable Analysis and Signal Separation*, ser. Lecture Notes in Computer Science, vol. 9237, Springer Berlin / Heidelberg, 2015, pp. 3–13.

[63] ——, “The concept of tensorization”, *Technical Report 17–99, ESAT-STADIUS, KU Leuven, Leuven, Belgium*, 2017.

[64] O. Debals, M. Sohail, and L. De Lathauwer, “Analytical multi-modulus algorithms based on coupled canonical polyadic decompositions”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 16-150, 2016.

[65] O. Debals, M. Van Barel, and L. De Lathauwer, “Löwner-based blind signal separation of rational functions with applications”, *IEEE Transactions on Signal Processing*, vol. 64, no. 8, pp. 1909–1918, 2016.

[66] W. Deburchgraeve, P. J. Cherian, M. D. Vos, R. M. Swarte, J. H. Blok, G. H. Visser, P. Govaert, and S. Van Huffel, “Neonatal seizure localization using PARAFAC decomposition”, *Clinical Neurophysiology*, vol. 120, no. 10, pp. 1787–1796, Oct. 2009.

[67] S. Diamond and S. Boyd, “Matrix-free convex optimization modeling”, *Springer Optimization and Its Applications*, pp. 221–264, 2016.

[68] W. Ding and Y. Wei, “Solving multilinear systems with $\mathcal{M}$-tensors”, *Journal of Scientific Computing*, vol. 68, no. 2, pp. 689–715, Aug. 2016.

[69] L.-M. Dogariu, S. Ciochină, J. Benesty, and C. Paleologu, “System identification based on tensor decompositions: A trilinear approach”, *Symmetry*, vol. 11, no. 4, 2019.

[70] I. Domanov and L. De Lathauwer, “On the uniqueness of the canonical polyadic decomposition of third-order tensors — Part I: Basic results and uniqueness of one factor matrix”, *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 3, pp. 855–875, Jul. 2013.

[71] ——, “On the uniqueness of the canonical polyadic decomposition of third-order tensors — Part II: Uniqueness of the overall decomposition”, *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 3, pp. 876–903, Jul. 2013.

[72] ——, “Canonical polyadic decomposition of third-order tensors: Reduction to generalized eigenvalue decomposition”, *SIAM Journal on Matrix Analysis and Applications*, vol. 35, no. 2, pp. 636–660, Apr. 2014.

[73] ——, “Generic uniqueness conditions for the canonical polyadic decomposition and INDSCAL”, *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 4, pp. 1567–1589, Nov. 2015.

[74] ——, “Generic uniqueness of a structured matrix factorization and applications in blind source separation”, *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 701–711, Jun. 2016.

[75] ——, “Canonical polyadic decomposition of third-order tensors: Relaxed uniqueness conditions and algebraic algorithm”, *Linear Algebra and its Applications*, vol. 513, pp. 342–375, Jan. 2017.

[76] ——, “On uniqueness and computation of the decomposition of a tensor into multilinear rank-$(1, L_r, L_r)$ terms”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 18-52, 2018.

[77] I. Domanov, A. Stegeman, and L. De Lathauwer, “On the largest multilinear singular values of higher-order tensors”, *SIAM Journal on Matrix Analysis and Applications*, vol. 38, no. 4, pp. 1434–1453, Jan. 2017.

[78] D. L. Donoho, “Compressed sensing”, *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[79] P. Dreesen, T. Goossens, M. Ishteva, L. De Lathauwer, and J. Schoukens, “Block-decoupling multivariate polynomials using the tensor block-term decompositions”, in *12th International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA 2015, Liberec, Czech Republic*, 2015, pp. 14–21.

[80] G. Eckart and G. Young, “The approximation of one matrix by another of lower rank”, *Psychometrika*, vol. 1, no. 3, pp. 211–218, Sep. 1936.

[81] Y. C. Eldar and G. Kutyniok, *Compressed sensing: Theory and applications.* Cambridge University Press, 2012.

[82] C. Elisei-Iliescu, C. Paleologu, J. Benesty, and S. Ciochina, “A recursive least-squares algorithm based on the nearest Kronecker product decomposition”, in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2019)*, May 2019, pp. 4843–4847.

[83] M. Espig, W. Hackbusch, T. Rohwedder, and R. Schneider, “Variational calculus with sums of elementary tensors of fixed rank”, *Numerische Mathematik*, vol. 122, no. 3, pp. 469–488, Nov. 2012.

[84] E. Evert and L. De Lathauwer, “Perturbation theory for CPD and for joint generalized eigenvalues”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-71, 2019.

[85] C. Févotte and J. Idier, “Algorithms for nonnegative matrix factorization with the β-divergence”, *Neural Computation*, vol. 23, no. 9, pp. 2421–2456, Sep. 2011.

[86] X. Gao, O. Edfors, F. Rusek, and F. Tufvesson, “Massive MIMO performance evaluation based on measured propagation data”, *IEEE Transactions on Wireless Communications*, vol. 14, no. 7, pp. 3899–3911, Mar. 2015.

[87] S. Geirnaert, G. Goovaerts, S. Padhy, M. Boussé, L. De Lathauwer, and S. Van Huffel, “Tensor-based ECG signal processing applied to atrial fibrillation detection”, in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, Oct. 2018, pp. 799–805.

[88] A. Georghiades, P. Belhumeur, and D. Kriegman, “From few to many: Illumination cone models for face recognition under variable lighting and pose”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 643–660, Jun. 2001.

[89] R. S. Ghiass and E. Fatemizadeh, “Multi-view face detection and recognition under varying illumination conditions by designing an illumination effect cancelling filter”, in *New Trends in Audio and Video / Signal Processing Algorithms, Architectures, Arrangements, and Applications SPA 2008*, Sep. 2008, pp. 27–32.

[90] N. Gillis, “The why and how of nonnegative matrix factorization”, in *Regularization, Optimization, Kernels, and Support Vector Machines*, ser. Machine Learning and Pattern Recognition, J. A. K. Suykens, M. Signoretto, and A. Argyriou, Eds., Chapman & Hall / CRC, 2014, ch. 12, pp. 257–291.

[91] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, “PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals”, *Circulation*, vol. 101, no. 23, e215–e220, Jun. 2000, Circulation Electronic Pages: http://circ.ahajournals.org/content/101/23/e215.full PMID:1085218.

[92] G. Goovaerts, M. Boussé, D. Do, L. De Lathauwer, and S. Van Huffel, “Analysis of changes in ECG morphology prior to in-hospital cardiac arrest using weighted tensor decompositions”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-44, 2019.

[93] G. Goovaerts, O. De Wel, B. Vandenberk, R. Willems, and S. Van Huffel, “Detection of irregular heartbeats using tensors”, in *Proceedings of the 42nd annual Conference of Computing in Cardiology (CinC 2015), Nice, France*, Sep. 2015, pp. 573–576.

[94] L. Grasedyck, “Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure”, *Computing*, vol. 72, no. 3–4, pp. 247–265, Jan. 2004.

[95] ——, “Polynomial approximation in hierarchical Tucker format by vector tensorization”, in, Preprint 43, DFG/SPP1324, RWTH Aachen, Apr. 2010.

[96] L. Grasedyck, D. Kressner, and C. Tobler, “A literature survey of low-rank tensor approximation techniques”, *GAMM-Mitteilungen*, vol. 36, no. 1, pp. 53–78, Feb. 2013.

[97] W. Guo, I. Kotsia, and I. Patras, “Tensor learning for regression”, *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 816–827, Feb. 2012.

[98] 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*, vol. 56, no. 7, pp. 3198–3213, Jul. 2008.

[99] W. Hackbusch, *Tensor spaces and numerical tensor calculus*. Springer, 2012, vol. 42.

[100] W. Hackbusch, D. Kressner, and A. Uschmajew, “Perturbation of higher-order singular values”, *SIAM Journal on Applied Algebra and Geometry*, vol. 1, no. 1, pp. 374–387, Jan. 2017.

[101] W. Hackbusch and A. Uschmajew, “On the interconnection between the higher-order singular values”, *Numerische Mathematik*, vol. 135, no. 3, pp. 875–894, Mar. 2017.

[102] N. Hao, M. E. Kilmer, K. Braman, and R. C. Hoover, “Facial recognition using tensor-tensor decompositions”, *SIAM Journal on Imaging Sciences*, vol. 6, no. 1, pp. 437–463, Jan. 2013.

[103] R. A. Harshman, “Determination and proof of minimum uniqueness conditions for PARAFAC1”, *UCLA Working Papers in Phonetics*, vol. 22, pp. 111–117, 1972.

[104] X. He, D. Cai, and P. Niyogi, “Tensor subspace analysis”, in *Advances in Neural Information Processing Systems*, Y. Weiss, B. Schölkopf, and J. C. Platt, Eds., vol. 18, MIT Press, 2006, pp. 499–506.

[105] R. Hennequin, B. David, and R. Badeau, “Beta-divergence as a subclass of Bregman divergence”, *IEEE Signal Processing Letters*, vol. 18, no. 2, pp. 83–86, Feb. 2011.

[106] G. Hollander, P. Dreesen, M. Ishteva, and J. Schoukens, “Approximate decoupling of multivariate polynomials using weighted tensor decomposition”, *Numerical Linear Algebra with Applications*, e2135–n/a,

[107] A. Holobar and D. Farina, “Blind source identification from the multichannel surface electromyogram”, *Physiological Measurement*, vol. 35, no. 7, R143, 2014.

[108] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge University Press, Cambridge, 2013.

[109] K. Huang and L. Zhang, “Cardiology knowledge free ECG feature extraction using generalized tensor rank one discriminant analysis”, *EURASIP Journal on Advances in Signal Processing*, vol. 2014, no. 1, p. 2, 2014.

[110] Y.-D. Huang and M. Barkat, “Near-field multiple source localization by passive sensor array”, *IEEE Transactions on Antennas and Propagation*, vol. 39, no. 7, pp. 968–975, Jul. 1991.

[111] B. Hunyadi, D. Camps, L. Sorber, W. Van Paesschen, M. De Vos, S. Van Huffel, and D. L. L., “Block term decomposition for modelling epileptic seizures”, *EURASIP Journal on Advances in Signal Processing*, vol. 2014, no. 139, pp. 1–19, Sep. 2014, Special Issue on Recent Advances in Tensor-Based Signal and Image Processing.

[112] B. Hunyadi, S. Van Huffel, and M. De Vos, “The power of tensor decompositions in biomedical applications”, in *Machine Learning for Healthcare Technologies*, D. A. Clifton, Ed., The Institution of Engineering and Technology (IET, London, UK), 2016.

[113] B. Hunyadi, W. Van Paesschen, M. De Vos, and S. Van Huffel, “Tensor decompositions and data fusion in epileptic EEG and fMRI data”, *WIREs Data mining and knowledge discovery*, pp. 1–15, Nov. 2016.

[114] A. Hyvärinen and E. Oja, “Independent component analysis: Algorithms and applications”, *Neural Networks*, vol. 13, no. 4–5, pp. 411–430, Jun. 2000.

[115] M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer, “Best low multilinear rank approximation of higher-order tensors, based on Riemannian trust-region scheme”, *SIAM Journal on Matrix Analysis and Applications*, vol. 32, no. 1, pp. 115–135, Jan. 2011.

[116] 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 geometric Newton method”, in *AIP Conference Proceedings*, vol. 1048, 2008, pp. 274–277.

[117] D. Jäckel, U. Frey, M. Fiscella, and A. Hierlemann, “Blind source separation for spike sorting of high-density microelectrode array recordings”, in *Proceedings of the 5th International IEEE EMBS Conference on Neural Engineering (NER) (Cancun, Mexico)*, Apr. 2011, pp. 5–8.

[118] W. C. Jakes, *Microwave mobile communications.* New York: Wiley, 1974.

[119] C. J. James and C. W. Hesse, “Independent component analysis for biomedical signals”, *Physiological Measurement*, vol. 26, no. 1, R15–R39, 2005.

[120] C. R. Johnson, H. Šmigoc, and D. Yang, “Solution theory for systems of bilinear equations”, *Linear and Multilinear Algebra*, vol. 62, no. 12, pp. 1553–1566, 2014.

[121] I. Jolliffe, *Principal Component Analysis.* Wiley Online Library, 2002.

[122] N. Kargas, N. Sidiropoulos, and X. Fu, “Tensors, learning, and “Kolmogorov extension” for finite-alphabet random vectors”, *IEEE Transactions on Signal Processing*, vol. 66, no. 18, pp. 4854–4868, Sep. 2018.

[123] B. N. Khoromskij, “$\mathcal{O}(d \log N)$-quantics approximation of $N-d$ tensors in high-dimensional numerical modeling”, *Constructive Approximation*, vol. 34, no. 2, pp. 257–280, Oct. 2011.

[124] T. G. Kolda, “Multilinear operators for higher-order decompositions”, Sandia National Laboratories, Albuquerque, NM, Livermore, CA, Tech. Rep., Apr. 2006, Tech. Report SAND2006-2081.

[125] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications”, *SIAM Review*, vol. 51, no. 3, pp. 455–500, Aug. 2009.

[126] W. P. Krijnen, T. K. Dijkstra, and A. Stegeman, "On the non-existence of optimal solutions and the occurrence of "degeneracy" in the CANDECOMP/PARAFAC model", *Psychometrika*, vol. 73, no. 3, pp. 431–439, Jan. 2008.

[127] H. Krim and M. Viberg, "Two decades of array signal processing: The parametric approach", *IEEE Signal Processing Magazine*, vol. 13, no. 4, pp. 67–94, 1996.

[128] J. B. Kruskal, "Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics", *Linear Algebra and its Applications*, vol. 18, no. 2, pp. 95–138, 1977.

[129] D. Kundur and D. Hatzinakos, "Blind image deconvolution", *IEEE Signal Processing Magazine*, vol. 13, no. 3, pp. 43–64, May 1996.

[130] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems", *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[131] M. Lewicki, "A review of methods for spike sorting: The detection and classification of neural action potentials", *Network: Computation in Neural Systems*, vol. 9, no. 4, pp. 53–78, 1998.

[132] Q. Li and D. Schonfeld, "Multilinear discriminant analysis for higher-order tensor data classification", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2524–2537, Dec. 2014.

[133] Y. Li, A. Cichocki, S. Amari, and S. Shishkin, "Analysis of sparse representation and blind source separation", *Neural computation*, vol. 16, no. 6, pp. 1193–1234, Jun. 2004.

[134] L.-H. Lim and P. Comon, "Nonnegative approximations of nonnegative tensors", *Journal of Chemometrics*, vol. 23, no. 7–8, pp. 432–441, 2009.

[135] ——, "Blind multilinear identification", *IEEE Transactions on Information Theory*, vol. 60, no. 2, pp. 1260–1280, Feb. 2014.

[136] S. Liu and G. Trenkler, "Hadamard, Khatri–Rao, Kronecker and other matrix products", *International Journal of Systems Science*, vol. 4, no. 1, pp. 160–177, 2008.

[137] L. Ljung, *System identification: Theory for the user*, second edition. Prentice hall, 1999.

[138] M. Llamedo, A. Khawaja, and J. Martínez, "Analysis of 12-lead classification models for ECG classification", in *2010 Computing in Cardiology*, Sep. 2010, pp. 673–676.

[139] H. Lütkepohl, *New introduction to multiple time series analysis*. Springer, 2005.

[140] M. Mahoney, M. Maggioni, and P. Drineas, "Tensor-CUR decompositions for tensor-based data", *SIAM Journal of Matrix Analysis and Applications*, vol. 30, no. 3, pp. 957–987, 2008.

[141] S. Miron, Y. Song, D. Brie, and K. Wong, "Multilinear direction finding for sensor-array with multiple scales of invariance", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 2057–2070, Jul. 2015.

[142] F. Miwakeichi, E. Martínez–Montes, P. A. Valdéss–Sosa, N. Nishiyama, H. Mizuhara, and Y. Yamaguchi, "Decomposing EEG data into space-time-frequency components using parallel factor analysis", *NeuroImage*, vol. 22, pp. 1035–1045, Jul. 2004.

[143] G. Monchen, B. Sinquin, and M. Verhaegen, "Recursive Kronecker-based vector autoregressive identification for large-scale adaptive optics", *IEEE Transactions on Control Systems Technology*, pp. 1–8, 2018, (accepted for publication).

[144] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*, 5th ed. John Wiley & Sons, 2012.

[145] M. Mørup, L. K. Hansen, C. S. Herrmann, J. Parnas, and S. M. Arnfred, "Parallel factor analysis as exploratory tool for wavelet transformed event-related EEG", *NeuroImage*, vol. 29, no. 3, pp. 938–947, Feb. 2006.

[146] D. Muti and S. Bourennane, "Survey on tensor signal algebraic filtering", *Signal Processing*, vol. 87, no. 2, pp. 237–249, Feb. 2007.

[147] J. Nagy and M. Kilmer, "Kronecker product approximation for preconditioning in three-dimensional imaging applications", *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 604–613, 2006.

[148] H. V. Nguyen and L. Bai, "Cosine similarity metric learning for face verification", in *10th Asian Conference on Computer Vision (ACCV 2010, Queenstwon, New Zealand)*, R. Kimmel, R. Klette, and A. Sugimoto, Eds., Springer Berlin Heidelberg, Nov. 2010, pp. 709–720.

[149] D. Nion and N. D. Sidiropoulos, "Adaptive algorithms to track the PARAFAC decomposition of a third-order tensor", *IEEE Transactions on Signal Processing*, vol. 57, no. 6, pp. 2299–2310, Jun. 2009.

[150] J. Nocedal and S. Wright, *Numerical optimization*. Springer New York, Jul. 2006.

[151] P. M. R. de Oliveira and V. Zarzoso, "Temporal stability of block term decomposition in noninvasive atrial fibrillation analysis", in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, Oct. 2018, pp. 816–820.

[152] I. Oseledets, "Tensor-train decomposition", *SIAM Journal of Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, Sep. 2011.

[153] I. Oseledets, D. V. Savostianov, and E. Tyrtyshnikov, "Tucker dimensionality reduction of three-dimensional arrays in linear time", *SIAM Journal of Matrix Analysis and Applications*, vol. 30, no. 3, pp. 939–956, 2008.

[154] P. Paatero, "A weighted non-negative least squares algorithm for three-way "PARAFAC" factor analysis", *Chemometrics and Intelligent Laboratory Systems*, vol. 38, pp. 223–242, Oct. 1997.

[155] S. Padhy, G. Goovaerts, M. Boussé, L. De Lathauwer, and S. Van Huffel, "The power of tensor-based approaches in cardiac applications", in *Biomedical Signal Processing - Advances in Theory, Algorithms, and Applications*, G. Naik, Ed., Accepted, Springer, 2019.

[156] C. Paleologu, J. Benesty, and S. Ciochina, "Linear system identification based on a Kronecker product decomposition", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, May 2018.

[157] J.-M. Papy, L. De Lathauwer, and S. Van Huffel, "Exponential data fitting using multilinear algebra: The decimative case", *Journal of Chemometrics*, vol. 23, pp. 341–351, Feb. 2009, (www.interscience.wiley.com).

[158] A.-H. Phan, P. Tichavský, and A. Cichocki, "Low complexity damped Gauss–Newton algorithms for CANDECOMP/PARAFAC", *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 1, pp. 126–147, Jan. 2013.

[159] L. N. Ribeiro, A. L. F. de Almeida, and J. C. M. Mota, "Identification of separable systems using trilinear filtering", in *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2015, Cancun, Mexico)*, Dec. 2015, pp. 189–192.

[160] R. Roy and T. Kailath, "ESPRIT-estimation of signal parameters via rotational invariance techniques", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 7, pp. 984–995, 1989.

[161] B. Rubehn, C. Bosman, R. Oostenveld, P. Fries, and T. Stiegliz, “A MEMS-based flexible multichannel ECoG-electrode array”, *Journal of Neural Engineering*, vol. 6, no. 3, pp. 1–10, 2009.

[162] B. Savas and L. Eldén, “Handwritten digit classification using higher order singular value decomposition”, *Pattern Recognition*, vol. 40, no. 3, pp. 993–1003, Mar. 2007.

[163] D. Seo, J. M. Carmena, J. M. Rabaey, E. Alon, and M. M. Maharbiz, “Neural dust: An ultrasonic, low power solution for chronic brain-machine interfaces”, *arXiv preprint arXiv:1307.2196*, Jul. 2013.

[164] T.-J. Shan, M. Wax, and T. Kailath, “On spatial smoothing for direction-of-arrival estimation of coherent signals”, *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, no. 4, pp. 806–811, Aug. 1985.

[165] N. Sidiropoulos, “Generalizing Carathéodory’s uniqueness of harmonic parametrization to N dimensions”, *IEEE Transactions on Information Theory*, vol. 47, no. 4, pp. 1687–1690, May 2001.

[166] N. D. Sidiropoulos and R. Bro, “On the uniqueness of multilinear decomposition of N-way arrays”, *Journal of Chemometrics*, vol. 14, no. 3, pp. 229–239, May 2000.

[167] N. Sidiropoulos, R. Bro, and G. Giannakis, “Parallel factor analysis in sensor array processing”, *IEEE Transactions on Signal Processing*, vol. 48, no. 8, pp. 2377–2388, Aug. 2000.

[168] N. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. Papalexakis, and C. Faloutsos, “Tensor decomposition for signal processing and machine learning”, *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, Jul. 2017.

[169] N. Sidiropoulos and G. Dimic, “Blind multiuser detection in W-CDMA systems with large delay spread”, *IEEE Signal Processing Letters*, vol. 8, no. 9, pp. 87–89, Mar. 2001.

[170] N. Sidiropoulos, G. Giannakis, and R. Bro, “Blind PARAFAC receivers for DS-CDMA systems”, *IEEE Transactions on Signal Processing*, vol. 48, no. 3, pp. 810–823, Mar. 2000.

[171] N. Sidiropoulos, E. E. Papalexakis, and C. Faloutsos, “Parallel randomly compressed cubes: A scalable distributed architecture for big tensor decomposition”, *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 57–70, Sep. 2014.

[172] V. de Silva and L.-H. Lim, “Tensor rank and the ill-posedness of the best low-rank approximation problem”, *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1084–1127, Sep. 2008.

[173] B. Sinquin and M. Verhaegen, “K4SID: Large-scale subspace identification with Kronecker modeling”, *IEEE Transactions on Automatic Control*, May 2018.

[174] ——, “QUARKS: Identification of large-scale Kronecker vector-autoregressive models”, *IEEE Transactions on Automatic Control*, 2018.

[175] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks”, *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, Jul. 2009.

[176] A. J. Sommese and C. W. Wampler II, *The numerical solution of systems of polynomials arising in engineering and science.* World Scientific, Hackensack, NJ, 2005.

[177] L. Sorber, M. Van Barel, and L. De Lathauwer, “Unconstrained optimization of real functions in complex variables”, *SIAM Journal on Optimization*, vol. 22, no. 3, pp. 879–898, Jul. 2012.

[178] ——, “Structured data fusion”, *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 586–600, Jun. 2015.

[179] ——, *Complex optimization toolbox v1.* Feb. 2013.

[180] ——, "Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank-$(L_r, L_r, 1)$ terms, and a new generalization", *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 695–720, Apr. 2013.

[181] M. Sørensen and L. De Lathauwer, "Blind signal separation via tensor decomposition with Vandermonde factor: Canonical polyadic decomposition", *IEEE Transactions on Signal Processing*, vol. 61, no. 22, pp. 5507–5519, Nov. 2013.

[182] ——, "Multidimensional harmonic retrieval via coupled canonical polyadic decomposition", *Technical Report 13-240, ESAT-STADIUS, KU Leuven, Belgium*, 2013.

[183] ——, "Coupled tensor decompositions in array processing", *Technical Report 13–241, ESAT-STADIUS, KU Leuven, Leuven, Belgium*, 2014.

[184] ——, "Coupled canonical polyadic decompositions and (coupled) decompositions in multilinear rank-$(L_{r,n}, L_{r,n}, 1)$ terms — Part I: Uniqueness", *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 2, pp. 496–522, Apr. 2015.

[185] M. Sørensen, L. De Lathauwer, P. Comon, S. Icart, and L. Deneire, "Canonical polyadic decomposition with orthogonality constraints", *SIAM Journal on Matrix Analysis and Applications*, vol. 33, no. 4, pp. 1190–1213, Oct. 2012.

[186] M. Sørensen, I. Domanov, and L. De Lathauwer, "Coupled canonical polyadic decompositions and (coupled) decompositions in multilinear rank-$(L_{r,n}, L_{r,n}, 1)$ terms — Part II: Algorithms", *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 3, pp. 1015–1045, Apr. 2015.

[187] M. Sørensen, F. Van Eeghem, and L. De Lathauwer, "Blind multichannel deconvolution and convolutive extensions of canonical polyadic and block term decompositions", *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 4132–4145, Aug. 2017.

[188] K. Tiels, M. Schoukens, and J. Schoukens, "Generation of initial estimates for Wiener-Hammerstein models via basis function expansions", *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 481–486, Aug. 2014.

[189] G. Tomasi and R. Bro, "PARAFAC and missing values", *Chemometrics and Intelligent Laboratory Systems*, vol. 75, no. 2, pp. 163–180, Feb. 2005.

[190] L. N. Trefethen and D. Bau, *Numerical Linear Algebra.* SIAM, 1997.

[191] J. Treichler and B. Agee, "A new approach to multipath correction of constant modulus signals", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, no. 2, pp. 459–472, Apr. 1983.

[192] M. G. Tsipouras, D. I. Fotiadis, and D. Sideris, "An arrhythmia classification system based on the RR-interval signal", *Artificial Intelligence in Medicine*, vol. 33, no. 3, pp. 237–250, 2005.

[193] S. Tu, R. Boczar, M. Simchowitz, M. Soltanolkotabi, and B. Recht, "Low-rank solutions of linear matrix equations via Procrustes flow", in *Proceedings of the International Conference on Machine Learning (ICML 2016, New York, USA)*, 2016.

[194] M. Turk and A. Pentland, "Eigenfaces for recognition", *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, Jan. 1991.

[195] E. E. Tyrtyshnikov, "How bad are Hankel matrices?", *Numerische Mathematik*, vol. 67, pp. 261–269, 2 Mar. 1994.

[196] F. Van Eeghem, M. Sørensen, and L. De Lathauwer, "Tensor decompositions with several block-Hankel factors and application in blind system identification", *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 4090–4101, Aug. 2017.

[197] S. Van Eyndhoven, M. Boussé, B. Hunyadi, L. De Lathauwer, and S. Van Huffel, "Single-channel EEG classification by multi-channel tensor subspace learning and regression", in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, Sep. 2018, pp. 1–6.

[198] C. Van Loan and N. Pitsianis, *Approximation with Kronecker Products*, M. Moonen, G. Golub, and B. De Moor, Eds. Dordrecht: Springer Netherlands, 1993, pp. 293–314.

[199] M. Vandecappelle, M. Boussé, N. Vervliet, and L. De Lathauwer, "CPD updating using low-rank weights", in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018, Calgary, Alberta, Canada)*, Apr. 2018, pp. 6368–6372.

[200] M. Vandecappelle, N. Vervliet, and L. De Lathauwer, "Canonical polyadic decomposition with general cost functions using generalized Gauss–Newton", ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-05, 2019.

[201] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen, "A new truncation strategy for the higher-order singular value decomposition", *SIAM Journal on Scientific Computing*, vol. 34, no. 2, A1027–A1052, Jan. 2012.

[202] M. A. O. Vasilescu, "Multilinear projection for face recognition via canonical decomposition", in *Face and Gesture 2011 (FG '11, Santa Barbara, CA)*, Mar. 2011, pp. 476–483.

[203] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces", in *Proceedings of the European Conference on Computer Vision (ECCV '02, Copenhagen, Denmark)*, May 2002, pp. 447–460.

[204] ——, "Multilinear image analysis for facial recognition", in *Object recognition supported by user interaction for service robots*, vol. 2, Aug. 2002, pp. 511–514.

[205] ——, "TensorTextures: Multilinear image-based rendering", *ACM Transactions on Graphics*, vol. 23, pp. 336–342, 2004.

[206] A.-J. van der Veen, "Algebraic methods for deterministic blind beamforming", *Proceedings of the IEEE*, vol. 86, no. 10, pp. 1987–2008, Oct. 1998.

[207] A.-J. van der Veen and A. Paulraj, "An analytical constant modulus algorithm", *IEEE Transactions on Signal Processing*, vol. 44, no. 5, pp. 1136–115, May 1996.

[208] N. Vervliet, "Compressed sensing approaches to large-scale tensor decompositions", PhD thesis, KU Leuven, May 2018.

[209] N. Vervliet and L. De Lathauwer, "A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors", *IEEE Journal on Selected Topics in Signal Processing*, vol. 10, pp. 284–295, 2 2016.

[210] ——, "Numerical optimization based algorithms for data fusion", in *Data Fusion Methodology and Applications*, M. Cocchi, Ed., Accepted, Elsevier, 2018.

[211] N. Vervliet, O. Debals, and L. De Lathauwer, "Tensorlab 3.0 — Numerical optimization strategies for large-scale constrained and coupled matrix/tensor factorization", in *Proceedings of the 50th Asilomar Conference on Signals, Systems and Computers (Pacific Grove, CA)*, Nov. 2016, pp. 1733–1738.

[212] ——, "Canonical polyadic decomposition of incomplete tensors with linearly constrained factors", *Technical Report 16—172, ESAT-STADIUS, KU Leuven, Leuven, Belgium*, 2017.

[213] ——, "Exploiting efficient representations in large-scale tensor decompositions", *SIAM Journal on Scientific Computing*, vol. 41, no. 2, A789–A815, Mar. 2019.

[214] N. Vervliet, O. Debals, L. Sorber, and L. De Lathauwer, “Breaking the curse of dimensionality using decompositions of incomplete tensors: Tensor-based scientific computing in big data analysis”, *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 71–79, Sep. 2014.

[215] N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer, *Tensorlab 3.0*, Mar. 2016.

[216] H. Wang and N. Ahuja, “Facial expression decomposition”, in *Proceedings Ninth IEEE International Conference on Computer Vision*, vol. 2, Oct. 2003, pp. 958–965.

[217] L. Y. Wang, G. G. Yin, J.-F. Zhang, and Y. Zhao, *System Identification with Quantized Observations*. Birkhäuser Boston, 2010.

[218] V. Zarzoso, “Parameter estimation in block term decomposition for noninvasive atrial fibrillation analysis”, in *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2017, Curaçao, Dutch Antilles)*, Dec. 2017, pp. 1–5.

[219] V. Zarzoso and P. Comon, “Optimal step-size constant modulus algorithm”, *IEEE Transactions on Communications*, vol. 56, no. 1, Jan. 2008.

[220] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, “Face recognition: A literature survey”, *ACM Computing Surveys*, vol. 35, no. 4, pp. 399–458, Dec. 2003.

[221] H. Zhou, L. Li, and H. Zhu, “Tensor regression with applications in neuroimaging data analysis”, *Journal of the American Statistical Association*, vol. 108, no. 502, pp. 540–552, Jan. 2013.

[222] M. Zibulevsky and B. A. Pearlmutter, “Blind source separation by sparse decomposition in a signal dictionary”, *Neural computation*, vol. 13, no. 4, pp. 863–882, Apr. 2001.

[223] R. Zink, B. Hunyadi, S. Van Huffel, and M. De Vos, “Tensor-based classification of an auditory mobile BCI without a subject-specific calibration phase”, *Journal of Neural Engineering*, vol. 13, no. 2, p. 026 005, Mar. 2016.

Curriculum Vitae

Martijn Boussé was born in Hasselt on July 4th, 1991. In 2014, he received the Master of Science in Mathematical Engineering, magna cum laude, from KU Leuven, Belgium. His master thesis was titled *Tensor-based algorithms for blind system identification*, supervised by prof. Lieven De Lathauwer. He started a PhD in September 2014 at the STADIUS Center of Dynamical Systems, Signal Processing and Data Analytics of the Electrical Engineering Department (ESAT), KU Leuven, under the supervision of prof. Lieven De Lathauwer. His research concerns the development of explicit and implicit tensor decomposition-based algorithms for blind signal separation, (blind) system identification, and pattern recognition. His research has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC Advanced Grant: BIOTENSORS (no. 339804), Fonds de la Recherche Scientifique – FNRS and Fonds Wetenschappelijk Onderzoek – Vlaanderen under EOS Project no. 30468160 (SeLMA), and Research Council KU Leuven: C1 project C16/15/059-nD. In 2018, he was a visiting researcher at the University of Virginia, Charlottesville, and University of Minnesota, Minneapolis with prof. Nikos Sidiropoulos. He received a travel grant from the Flanders Research Foundation (FWO) to support this research. He received a Best Poster Award at the 25th Belgian-Dutch Conference on Machine Learning and was nominated for a Best Paper Award at the 2017 IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing.

List of publications

Journal papers

[1] **M. Boussé**, N. Vervliet, I. Domanov, O. Debals, and L. De Lathauwer, “Linear systems with a canonical polyadic decomposition constrained solution: Algorithms and applications”, *Numerical Linear Algebra with Applications*, vol. 25, no. 6, e2190, Aug. 2018.

[2] **M. Boussé**, O. Debals, and L. De Lathauwer, “Tensor-based large-scale blind system identification using segmentation”, *IEEE Transactions on Signal Processing*, vol. 65, no. 21, pp. 5770–5784, Nov. 2017.

[3] **M. Boussé**, O. Debals, and L. De Lathauwer, “A tensor-based method for large-scale blind source separation using segmentation”, *IEEE Transactions on Signal Processing*, vol. 65, no. 2, pp. 346–358, Jan. 2017.

Book chapters

[1] S. Padhy, G. Goovaerts, **M. Boussé**, L. De Lathauwer, and S. Van Huffel, “The power of tensor-based approaches in cardiac applications”, in *Biomedical Signal Processing - Advances in Theory, Algorithms, and Applications*, G. Naik, Ed., Accepted, Springer, 2019.

Conference proceedings

[1] **M. Boussé** and L. De Lathauwer, “Nonlinear least squares algorithm for canonical polyadic decomposition using low-rank weights”, in *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2017, Curaçao, Dutch Antilles)*, Dec. 2017, pp. 39–43.

[2] **M. Boussé** and L. De Lathauwer, “Large-scale autoregressive system identification using Kronecker product equations”, in *2018 6th IEEE Global Conference on Signal and Information Processing (GlobalSIP 2018, Anaheim, California, USA)*, Nov. 2018, pp. 1348–1352.

[3] **M. Boussé**, O. Debals, and L. De Lathauwer, “A novel deterministic method for large-scale blind source separation”, in *Proceedings of the 23rd European Signal Processing Conference (EUSIPCO 2015, Nice, France)*, Aug. 2015, pp. 1935–1939.

[4] **M. Boussé**, O. Debals, and L. De Lathauwer, “A tensor-based method for large-scale blind system identification using segmentation”, in *Proceedings of the 24th European Signal Processing Conference (EUSIPCO 2016, Budapest, Hungary)*, Sep. 2016, pp. 2015–2019.

[5] **M. Boussé**, G. Goovaerts, N. Vervliet, O. Debals, S. Van Huffel, and L. De Lathauwer, “Irregular heartbeat classification using Kronecker product equations”, in *39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2017, Jeju Island, South-Korea)*, Jul. 2017, pp. 438–441.

[6] **M. Boussé**, N. Vervliet, O. Debals, and L. De Lathauwer, “Face recognition as a Kronecker product equation”, in *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP 2017, Curaçao, Dutch Antilles)*, Dec. 2017, pp. 276–280.

[7] S. Geirnaert, G. Goovaerts, S. Padhy, **M. Boussé**, L. De Lathauwer, and S. Van Huffel, “Tensor-based ECG signal processing applied to atrial fibrillation detection”, in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, Oct. 2018, pp. 799–805.

[8] S. Van Eyndhoven, **M. Boussé**, B. Hunyadi, L. De Lathauwer, and S. Van Huffel, “Single-channel EEG classification by multi-channel tensor subspace learning and regression”, in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, Sep. 2018, pp. 1–6.

[9] M. Vandecappelle, **M. Boussé**, N. Vervliet, and L. De Lathauwer, “CPD updating using low-rank weights”, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018, Calgary, Alberta, Canada)*, Apr. 2018, pp. 6368–6372.

[10] M. Vandecappelle, **M. Boussé**, N. Vervliet, M. Vendeville, R. Zink, and L. De Lathauwer, “Tensorlab 4.0 – a preview”, in *14th International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA 2018, Guildford, UK)*, Jul. 2018, pp. 1–2.

Reports

[1] **M. Boussé**, I. Domanov, and L. De Lathauwer, “Linear systems with a multilinear singular value decomposition constrained solution”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 17-56, 2017.

[2] **M. Boussé**, N. Sidiropoulos, and L. De Lathauwer, “NLS algorithm for Kronecker-structured linear systems with a CPD constrained solution”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-13, 2019, Accepted for publication in the proceedings of the 27th European Signal Processing Conference (EUSIPCO 2019, A Coruña, Spain).

[3] G. Goovaerts, **M. Boussé**, D. Do, L. De Lathauwer, and S. Van Huffel, “Analysis of changes in ECG morphology prior to in-hospital cardiac arrest using weighted tensor decompositions”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-44, 2019.

[4] G. Goovaerts, S. Padhy, **M. Boussé**, L. De Lathauwer, and S. Van Huffel, “The power of tensor-based approaches in ECG signal processing”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-46, 2019, Accepted for publication in Biomedical data fusion using tensors of the 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2019, Berlin, Germany).

[5] N. Vervliet, M. Vandecappelle, **M. Boussé**, R. Zink, and L. De Lathauwer, “Recent numerical and conceptual advances for tensor decompositions — A preview of Tensorlab 4.0”, ESAT-STADIUS, KU Leuven, Leuven, Belgium, Tech. Rep. 19-45, 2019, Accepted for publication in IEEE Data Science Workshop (DSW 2019, Minneapolis, USA).

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
STADIUS CENTER FOR DYNAMICAL SYSTEMS, SIGNAL PROCESSING AND DATA ANALYTICS
Kasteelpark Arenberg 10 - box 2446
B-3001 Leuven
Martijn.Bousse@kuleuven.be
http://www.esat.kuleuven.be/stadius/