Université de Mons
Faculté polytechnique
Mathématique et recherche opérationnelle

Nonnegative Matrix and Tensor Factorizations: Models, Algorithms and Applications

Andersen Man Shun Ang

A thesis submitted
in partial fulfillment of the requirements for the degree of
Doctorat en Science de l'ingénieur et technologie

Dissertation committee:

Prof. Xavier Siebert	Université de Mons (Chair)
Prof. Nicolas Gillis	Université de Mons (Supervisor)
Prof. Thierry Dutoit	Université de Mons
Prof. Laurent Jacques	UCLouvain
Prof. Francois Glineur	UCLouvain
Prof. Lieven De Lathauwer	KULeuven

September, 2020

Abstract

In many fields, such as linear algebra, computational geometry, combinatorial optimization, analytical chemistry and geoscience, nonnegativity of the solution is required, which is either due to the fact that the data is physically nonnegative, or that the mathematical modeling of the problem requires nonnegativity. Image and audio processing are two examples for which the data are physically nonnegative. Probability and graph theory are examples for which the mathematical modeling requires nonnegativity.

This thesis is about the nonnegative factorization of matrices and tensors: namely nonnegative matrix factorization (NMF) and nonnegative tensor factorization (NTF). NMF problems arise in a wide range of scenarios such as the aforementioned fields, and NTF problems arise as a generalization of NMF. As the title suggests, the contributions of this thesis are centered on NMF and NTF over three aspects: modeling, algorithms and applications.

On the modeling aspect, we study two specific classes of NMF problems, namely the Minimum-volume NMF (minvol NMF) and the Nonnegative Unimodal Matrix Factorization (NuMF). Minvol NMF generalizes other classes of NMF problems and it can be shown that it leads to identifiability under some mild conditions, that is, the solution of minvol NMF is unique. On the NuMF, we provide an efficient algorithm for solving the problem. Both minvol NMF and NuMF are then applied on real-world datasets to demonstrate their effectiveness on solving some real-world machine learning tasks, namely hyperspectral imaging, audio blind source separation and analytical chemistry.

On the algorithmic side, we improve existing algorithms for solving NMF and NTF problems by introducing an acceleration framework, namely the Heuristic Extrapolation with Restarts (HER). Being a general acceleration framework, HER can be used to accelerate various Block Coordinate Descent (BCD) methods for solving NMF and NTF problems. The effectiveness of HER on accelerating the convergence of various BCDs is illustrated by experiments on synthetic and real datasets under different experimental settings.

On the application side, we used minvol NMF on hyperspectral imaging and audio source separation problems, and NuMF on chemistry data to demonstrate that NMF can produce meaningful decomposition of nonnegative data.

Acknowledgment

I am dumb and I know too little, and hence this Ph.D. thesis could not have been possible without the help of many people that I would like to thank. I would like to thank my supervisor Prof. Nicolas Gillis, for the patience, advice, guidance and supports on my research work during the past 3.5 years since I was lucky enough to be his first Ph.D. student. His suggestions were always insightful and helped me a lot to become a better researcher. I would also like to thank Dr. Jeremy E. Cohen and Dr. L. T. K. Hien for their helpful suggestions on my works. Then, I would like to thank my collaborator Valentin Leplat for our fruitful collaborations and discussions. Special thanks go to Dr. Arnaud Vandaele for proofreading my thesis. Finally, I would like to thank my colleagues in the COLORMAP group, including the former and current members: Dr. Arnaud Vandaele, Dr. Punit Sharma, Mr. Valentin Leplat, Dr. L. T. K. Hien, Dr. Junjun Pan, Mr. Nicolas Nadisic, Mr. Francois Moutier, Dr. Timothy Marrinan, Mr. Pierre De Handschutter, Dr. Christophe Kervazo, and Dr. Maryam Abdolali, for them to wandering around with me in during my study in Mons.

I would also like to thank Dr. Ting Kei Pong at the Hong Kong Polytechnic University, Dr. Jeremy E. Cohen at the University of Rennes and Prof. Hans De Sterck at the University of Waterloo for hosting me.

Last but not least, I would like to thank my mum for her love.

Contents

List of Figures

List of Tables

1 Introduction

> In mathematics you don't understand things. You just get used to them.
>
> *John von Neumann*

This thesis is about the nonnegative factorization of matrices and tensors: the Nonnegative Matrix Factorization (NMF) and the Nonnegative Tensor Factorization (NTF). NMF problems arise in a wide range of scenarios, such as linear algebra, computational geometry, combinatorial optimization, analytical chemistry and geoscience, while NTF arises as a generalization of NMF. In many applications of the aforementioned fields, the nonnegativity of solution is required, which is either due to the fact that (i) the data is physically nonnegative, or (ii) the mathematical modeling of the problem requires nonnegativity. Image and audio processing are examples for which the data are physically nonnegative. Probability and graph theory are examples for which the mathematical modeling requires nonnegativity.

1.1 A short introduction to NMF and NTF

As factorization problems, both NMF and NTF share the same goal: given a dataset (a matrix or a tensor), find a set of factors fulfilling the nonnegative constraints whose product fits the data. Given a matrix $\mathbf{M}$, the goal of NMF is to find two nonnegative factor matrices $\mathbf{W}$ and $\mathbf{H}$ such that their product equals to, or approximates, $\mathbf{M}$. In general, the matrices $\mathbf{M}$ can be complex-valued or even quaternion-valued [37], but in this thesis, we focus on real matrices. NTF generalizes the notion of NMF from matrix to tensors, and the problem in NTF is similar to NMF: given a tensor $\mathcal{T}$, find nonnegative factor matrices such that their product equals to $\mathcal{T}$.

NMF and NTF problems do not have closed-form solution in general. In fact, NMF and NTF are computationally intractable, that is (i.e.), they are NP-Hard problems [27, 59, 106]. As it is not expected to find a polynomial-time method to solve exactly general NMF/NTF problems, practitioners usually look for computationally efficient heuristics to solve approximately NMF/NTF. These heuristics, instead of finding factors that give an exact decomposition of the input data, find factors that give a good approximation of the data, where the quality of the approximation, or in other words the quality of fit, is quantified by some particular choice of distance measures. In general, non-metric distance such as divergence functions can be used to quantify the quality of the approximation. In this thesis, we focus mostly on the Euclidean distance, i.e., the Frobenius norm, which corresponds to the maximum likelihood estimator in the presence of independent and identically distributed (i.i.d.) Gaussian noise. However, note that Gaussian noise becomes less appropriate when the nonnegative data is becoming sparser. In this thesis, we deal mostly with dense data and therefore Gaussian noise is a reasonable assumption.

Two main concerns with NMF and NTF:

- The first is to make the problem identifiable by modifying it. Identifiability refers to the uniqueness of the solution when solving the problem. A way to make the problem identifiable is to impose additional conditions or constraints on top of the original models. In this thesis, we discuss two of such particular conditions on NMF, namely the minimum-volume and the unimodal conditions. We will study how these conditions affect NMF. In particular, how these additional conditions help in obtaining identifiability results.

- An identifiable problem can be very difficult to solve. In fact, NMF problem itself is NP-hard. Hence, the second issue here is to design heuristics to solve the problem. A heuristic is not guaranteed to find the optimal solution, but it can find good solutions efficiently and is therefore useful. In this thesis, we introduce a generic framework called "HER" for accelerating various algorithms that are used to solve NMF/NTF problems. The framework is an extrapolation-based scheme that can speed up empirically various algorithms solving NMF and NTF problem in various settings, while the framework has a low extra computational overhead compared to the original algorithm.

A more technical and in-depth introduction to NMF and NTF will be given in §1.4 and §6, respectively. Now we discuss about the thesis itself. We begin by summing up the contributions of this thesis in the following section.

1.2 Contribution and thesis organization

As the name suggests, the contributions of this thesis are centered on NMF and NTF over three areas: modeling, algorithms and application aspects, as detailed below.

1. **Modeling aspect: minvol NMF and NuMF**
    - We study a specific class of NMF problem called *Minimum-Volume NMF* (minvol NMF). This class of NMF problem generalizes another class of NMF called *Separable NMF*. We perform comparison studies on the choice of volume function in minvol NMF, and provide an identifiability theorem on one particular minvol NMF model that uses determinant volume, and we argue that such a minvol NMF model is superior than other existing minvol NMF approaches; see §2.
    - We study a specific class of NMF problem called *Nonnegative Unimodal Matrix Factorization* (NuMF). We provide an efficient algorithm for solving NuMF, using a series of techniques: multi-grid, efficient algorithm design, and acceleration. We also three preliminary results on the identifiability of NuMF under three special cases. See §5.
    - The papers related to this aspect are [6, 4, 71, 72, 73] .
2. **Algorithm aspect: HER acceleration framework**
    - We provide efficient algorithms for solving minvol NMF (see §2.2) and NuMF (see §5.1.7 and §5.2).
    - We introduce a generic framework, named *Heuristic Extrapolation with Restarts* (HER), for accelerating block coordinate descent type of algorithms solving NMF and NTF problems. We provide empirical evidence that HER can significantly improve the convergence of various existing BCD algorithms for the problems in various scenarios. See §7.
    - The papers related to this aspect are [1, 5, 3, 2] .
3. **Application aspect: Hyperspectral unmixing, audio source separation and analytical chemistry**

- We demonstrate the usefulness of NMF models in two applications: hyperspectral imaging (see §3) and audio source separation (see §4).
- We demonstrate the usefulness of NuMF in analytical chemistry application: for decomposing chemical spectral data for identifying the chemical component presented in the data. See §5.
- The papers related to this aspect are [6, 4, 71, 72, 73] .

This thesis is the ensemble of the following research efforts made by the author and his coauthors: three journal papers [6, 72, 5], five conference papers [4, 71, 73, 1, 2], one journal preprint [3] under review and one working paper on the content of §5. Many parts of the thesis material appear in these papers.

Thesis organization

- **Chapter** 1 **Introduction** We introduce the whole thesis, give a preview on the contributions and structure of the thesis. This chapter also serves as the technical backbone for Chapter 2 to Chapter 5 by presenting the basics on NMF. We lay-down some basic knowledge on NMF in §1.4. Then we go to a specific model called Separable NMF in §1.5 as an introductory material to the next chapter.
- **Chapter** 2 **NMF with minimum volume** We pay our attention to a generalization of Separable NMF called minimum-volume NMF (minvol NMF) in §2.1, where we discuss about the modeling issues and provide an identifiability theorem on minvol NMF. In §2.2, we briefly discuss how to solve minvol NMF. Finally, we conclude this chapter in §2.3, where we list some open problems related to minvol NMF. The material of this chapter appears in [6, 4, 71, 72, 73].
- **Chapter** 3 **Minvol NMF on hyperspectral unmixing** We discuss NMF applied to hyperspectral unmixing. We first give a brief introduction about hyperspectral unmixing in §3.1, then in §3.2 we discuss how algorithms are compared. Finally in §3.3 we present the numerical results on comparing NMF models presented in §2 with some other algorithms. The material of this chapter appears in [6, 4, 71, 72, 73].
- **Chapter** 4 **Minvol NMF on audio source separation** We discuss NMF on the application of audio *Blind Source Separation* (BSS). We treat NMF as a black box and we do not focus too deep on the details on the algorithm design here, rather, we focus on how NMF can be used to solve audio BSS problems. For the details on the NMF algorithms design, see [72, 73]. We first give a brief introduction about audio BSS in §4.1, then we look at how NMF can be used to solve audio BSS problem in §4.2, where we provide a few examples on decomposing piano recording. The material of this chapter appear in [72, 73].
- **Chapter** 5 **NMF with unimodality: NuMF** We pay attention to another specific NMF models, namely NuMF. We introduce NuMF and lay-down the foundation in §5.1, where we characterize the unimodal property and show that NuMF is a Mixed-Integer Programming (MIP) problem that is nonconvex and also block-nonconvex. To solve such MIP, we propose a simple but naive brute-force heuristics strategy, which is then improved by using the multi-grid

method as a dimension reduction step in §5.2, where we also show that the restriction operator preserves the unimodality. Then we present three preliminary results regarding the identifiability of NuMF in §5.3 under three special cases. We present empirical results concerning the algorithms and the theories on NuMF in §5.4 on synthetic and real datasets. Finally, we summarize this chapter in §5.5 and present some open problems.

- **Chapter** 6 **Tensor factorization** We discusses tensor factorization and the solution approach to it. We first give a review on the formalism of tensor for the purpose of this thesis in §6.1. We show that NMF is a special case of NTF in §6.2. Then we discuss the general solution approach to the NTF, CPD and NMF problems in §6.3. The material of this chapter appears in [1, 5, 3, 2].
- **Chapter** 7 **Heuristic extrapolation with restarts** We introduce an acceleration framework, namely HER, for accelerating the algorithms solving NTF. First we give the idea of acceleration through extrapolation in §7.1. Then, in §7.2, we present the original form of HER for NMF algorithms. Next, we present HER framework for general tensor problems in §7.3. After that, we present the numerical results of HER compared with other algorithms on NMF, NTF and CPD problems in §7.4, §7.5,§7.6, respectively. Finally, we summarize this chapter in §7.7, where we also present some open problems. The material of this chapter appears in [1, 5, 3, 2].
- **Chapter** 8 **Conclusion** Here we conclude the thesis by summarizing the findings of the previous chapters. We also restate the open problems related to the research conducted in the thesis.

1.3 Miscellaneous items concerning the thesis

In this section we list some remarks concerning the convention and setting used in the thesis.

Setting on numerical experiments The experiments presented in the papers by the author, as well as the experiments presented in this thesis, are mostly performed on `MATLAB` (v.2015a) under a single thread environment on a laptop computer with 2.4GHz CPU and 16GB RAM. Some of the codes are available at `https://angms.science/research.html`.

Notations and glossary As the thesis is a ensemble of multiple works by the author, the notations may not be consistent, especially for some figures appeared in the experimental sections. Efforts have been made on standardizing the notations. In general, we denote $\mathbf{M}$ (and sometimes $\mathbf{X}$) as the data matrix. Given a matrix $\mathbf{A}$, we let $\mathbf{A}(i,:)$ or $\mathbf{a}^i$ to denote the ith row of $\mathbf{A}$, and we let $\mathbf{A}(:,i)$ or $\mathbf{a}_i$ to denote the ith column of $\mathbf{A}$. We will introduce the notations for NMF in the remaining part in this chapter, and we introduce the notations for tensor in §6. At the end of the thesis, we attached a list of notation and glossary for quick referencing.

Figure, table, and convention Efforts have been made on making the figures in this thesis more visualizable, and all figures are best viewed in color. When listing the result in table form, the best results are bolded, and the worst results are denoted in red color.

The section symbol We abuse the symbol § to denote both the section number and the chapter number.

Convention on writing optimization problems In this thesis, we heavily abuse the notation in writing optimization problems. For example, for the problem of minimizing the function $f(\mathbf{x})$ that has the expression $\frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|_2$, i.e.,

$$\text{minimize } f(\mathbf{x}) \text{ with respect to } \mathbf{x} \text{ where } f(\mathbf{x}) = \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|_2,$$

will be written as

$$\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x}) \ := \ \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|_2,$$

or compactly as

$$\min_{\mathbf{x}} f(\mathbf{x}) = \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|_2.$$

The prerequisites for reading this thesis To read this thesis smoothly, it is expected that the reader should be at least familiar with linear algebra and optimization. Experience in matrix factorization is not a must but it would be an advantage. It is also assumed the reader do not have knowledge in the application fields of NMF or NTF.

1.4 Technical introduction to NMF

Now, in the remaining parts of this chapter, we focus on NMF.

NMF is a popular research topic As stated in the beginning of this chapter, NMF problems arise in a wide range of scenarios. A simple search on Google Scholar with the key words “nonnegative matrix factorization” gives about 83400 results, while using the words “non-negative matrix factorization” gives about 110000 results. The research on NMF is huge and therefore the vast background of NMF makes it impossible to review all aspects of it. Furthermore, a simple search on Google Scholar with the key words “nonnegative matrix factorization + survey” gives more than 10 reviews or survey papers on NMF, for examples [99, 11, 30, 108, 98, 113, 44, 40, 75, 102, 57, 79, 111, 13] . There is a complete volume on such subject [21] . All these works show that NMF is a popular research topic, and justify doing research on NMF.

In this section, we give a more in-depth introduction to NMF, which is then used as the backbone for the next few chapters. However, we only focus on the background material of NMF that are the minimum necessity for the purpose of this thesis. For a comprehensive treatment of NMF, see the up-coming book [45] , or the surveys mentioned above.

Organization In this section, we first discuss the technical details related to NMF in §1.4.1 . Then we briefly introduce how NMF is used in application in §1.4.2 . Next we talk about how NMF problems are solved in §1.4.3 . We then give a few words on the uniqueness of the solution of NMF in §1.4.4, followed by the geometric interpretation of the solution in §1.4.5 . As NMF is in fact NP-hard problems (§1.4.6), a twist is often made on the model to make it tractable. We move on to such a specific NMF model called the Separable NMF in §1.5 .

1.4.1 NMF

Nonnegative Matrix Factorization (NMF) is a linear algebra problem: given a nonnegative matrix $\mathbf{M}$ and a factorization rank $r \in \mathbb{N}$, find two nonnegative matrices $\mathbf{W}$ and $\mathbf{H}$ such that $\mathbf{M} = \mathbf{WH}$. This is called the *exact-NMF*. If exactness is not required and we seek for an approximate in the decomposition, then the problem is *approximate-NMF*: given $\mathbf{M}$, find the pair $(\mathbf{W}, \mathbf{H})$ such that $\mathbf{M} \approxeq \mathbf{WH}$. While currently there is no exact method to solve a general NMF problem, we consider solving NMF using numerical optimization approaches by casting NMF problem as the following nonconvex optimization problem:

$$\text{NMF} : \underset{\mathbf{W},\mathbf{H}}{\text{minimize}} \ \frac{1}{2}\|\mathbf{M} - \mathbf{WH}\|_F^2 \ \text{ subject to } \ \mathbf{W} \geq 0, \mathbf{H} \geq 0, \tag{1.1}$$

where $\mathbf{M} \in \mathbb{R}_+^{m \times n}$ is the input matrix , $\mathbf{W} \in \mathbb{R}_+^{m \times r}$ and $\mathbf{H} \in \mathbb{R}_+^{r \times n}$ are the optimization variables, and the inequalities $\mathbf{W} \geq 0$ and $\mathbf{H} \geq 0$ mean that $\mathbf{W}$ and $\mathbf{H}$ are elementwise nonnegative.

In problem (1.1) , the hidden parameter r is called the *factorization rank*, which is in general an unknown parameter in NMF. It represents the embedding dimension of $\mathbf{M}$ in the column space of $\mathbf{W}$, or to be precise, the dimension of the conical hull of (the columns of) $\mathbf{W}$. In fact, this value is closely related to the *nonnegative rank* of the matrix $\mathbf{M}$, defined as

$$\text{rank}_+(\mathbf{M}) = \min \left\{ k \;\middle|\; \begin{array}{l} \sum_{i=1}^k \mathbf{A}_i = \mathbf{M}, \text{rank}\,(\mathbf{A}_1) = \cdots = \text{rank}\,(\mathbf{A}_k) = 1, \\ \mathbf{A}_1 \geq 0, \mathbf{A}_2 \geq 0, \ldots, \mathbf{A}_k \geq 0 \end{array} \right\},$$

which is NP-hard to compute [106]. Therefore determining the accurate value of r such that $\mathbf{M}$ has an exact NMF remains open. In fact, such a problem, called *Model Order Selection* in application domains, is a research topics on its own, and hence in this thesis we made the following (strong) assumption:

We assume the value of r is given when solving an NMF problem. (1.2)

Lastly, we mention that this assumption does not greatly deteriorate the value of using NMF on real-world applications due to the following reasons.

- In applications, most of the time we have some information on the range of values of r based on prior domain knowledge.

- In some cases, NMF algorithms can perform automatic model order selection. For example, when applying minvol NMF on the audio data, it is observed that when the value r is overestimated, these extra components in the final solution will be set to zero. We discuss minvol NMF further in §2.1 , and the automatic model order selection phenomenon in §2.1.3 . We give experimental verification of this phenomenon in §4.2 .

- As the last resort, we can always brute force on r: try several different values of r on NMF on the data, and pick the one with the best fit as the correct value.

For the recent progress on estimating the value of r (the nonnegative rank) in NMF, see [45, 31].

1.4.2 How NMF is used in applications

Being practically useful is one of the driving factor that motivates the study of NMF as NMF finds applications in many areas [73, 44, 81]. We illustrate two examples: unmixing applications in hyperspectral imaging and audio blind source separation; see Fig.1.1 .

Here we briefly talk about how NMF can be used in applications, using Fig.1.1 as examples. We begin with the data matrix $\mathbf{M} \in \mathbb{R}_+^{m \times n}$.

Data matrix As matrices are tabular data representation, a data matrix $\mathbf{M}$ can capture two data modalities in the columns and in the rows. The two modalities together give specific information about the data. In the first example in Fig.1.1 , the matrix $\mathbf{M}$ is a nonnegative spatial-wavelength matrix with two modalities: 1) the spatial coordinates that represent the actual location of the pixel in the image, and 2) the wavelength modality that presents the spectral signature of the pixel. In the second example in the figure, the matrix $\mathbf{M}$ is a nonnegative time-frequency matrix with two modalities: frequency and time. The two coordinates together indicate the distribution of the energy of the audio signal across time and frequency. We give more details on these datasets in §3 and §4 .

The factors $\mathbf{W}$ and $\mathbf{H}$ The interpretation of the factors $\mathbf{W}$ and $\mathbf{H}$ are associated to the column-modality and the row-modality of the data matrix $\mathbf{M}$, respectively. The common names for the interpretation of $\mathbf{W}$, if data points are represented as columns, are "basis", "dictionary", "atom" that more or less mean the "principle factor" that generates the data, and the common names for the interpretation of $\mathbf{H}$ are "coefficient", "weight", "activation", "abundance", which all mean the "amount of existence" of the basis in the data. For example, in the second example of Fig.1.1 , the columns in $\mathbf{W}$ are the frequency profile of the note extracted from the audio data, while the rows in $\mathbf{H}$ are the time activation profile of these notes in the audio data.

Fig.1.2 gives a pictorial description on how NMF is used in application, demonstrated using hyperspectral imaging as an example, where $\mathbf{w}_1$ denotes the first column of $\mathbf{W}$ and $\mathbf{h}^3$ denotes the third row of $\mathbf{H}$. We give some common interpretations about the vectors $\mathbf{w}_i, \mathbf{h}_i, \mathbf{h}^i$ in Table 1.1 .

Table 1.1: Common interpretations about the vectors $\mathbf{m}_i, \mathbf{w}_i, \mathbf{h}_i$ and $\mathbf{h}^i$.

Symbol	Meaning	Size	Interpretation
$\mathbf{m}_i$	ith column of $\mathbf{M}$	m-by-1	The ith data point.
$\mathbf{w}_i$	ith column of $\mathbf{W}$	m-by-1	The ith basis.
$\mathbf{h}^i$	ith row of $\mathbf{H}$	1-by-n	The activation of the ith basis in the whole dataset.
$\mathbf{h}_i$	ith column of $\mathbf{H}$	r-by-1	The amount of composition of $\mathbf{w}_1, \dots, \mathbf{w}_r$ in the ith data point.

1.4.3 How NMF problems are solved, BCD and HALS

Most algorithms for solving NMF problem (1.1) use a two-block coordinate descent (BCD) scheme by optimizing alternatively over $\mathbf{W}$ for $\mathbf{H}$ fixed and vice versa; see Algorithm 1 .

By symmetry, since $\|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F = \|\mathbf{M}^\top - \mathbf{H}^\top \mathbf{W}^\top\|_F$, the updates of $\mathbf{W}$ and $\mathbf{H}$ are usually based on the same strategy. The subproblem for $\mathbf{H}$ is a Nonnegative Least Squares (NNLS) problem:

$$\min_{\mathbf{H} \geq 0} \; \|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 \tag{1.3}$$

Algorithm 1 Framework for most NMF algorithms

```
Input: A matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ and a factorization rank $r$.
Output: An approximate solution $(\mathbf{W}, \mathbf{H})$ to NMF (1.1).
Initialization: $\mathbf{W} \in \mathbb{R}_+^{m \times r}$, $\mathbf{H} \in \mathbb{R}_+^{m \times r}$.
for $k = 1, \dots$ until some criteria is satisfied do
    $\mathbf{W} \leftarrow \operatorname*{argmin}_{\mathbf{W} \geq 0} \|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2$.    % Update W by solving a NNLS subproblem.
    $\mathbf{H} \leftarrow \operatorname*{argmin}_{\mathbf{H} \geq 0} \|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2$.    % Update H by solving a NNLS subproblem.
end for
```

that needs to be solved exactly or approximately.

We now review the common approaches in the NMF community to solve Problem (1.3), they are

- The multiplicative updates, first proposed in [26] and rediscovered in [67].
- The active-set methods; see [61] and the reference therein. These methods solve Problem (1.3) exactly, and the algorithm using active-set is named Alternating Nonnegative Least Squares (ANLS).
- The projected gradient method [78]. Such method solves Problem (1.3) using projected gradient steps. Recently, this line of research regains its popularity due to the advances in first-order optimization [10], and the method has been extended to block projected gradient methods; see [109, 54].
- The exact block coordinate descent (BCD) methods called Hierarchical Alternating Least Squares (HALS); proposed in [15, pp.161-170] and rediscovered in [20]. We discuss more on HALS below.

For details and comparisons between these methods, see [46]. Among these approaches, we emphasize that the popular multiplicative update is slow for the NMF problem (1.1), while HALS schemes have been shown to be very effective in many situations [46]. For this reason, we focus on HALS in this thesis, and we explain HALS below.

Consider the update of a single row of $\mathbf{H}$, denoted as $\mathbf{h}^j$, while the others rows are fixed. The minimization problem admits a simple closed-form solution: for all j,

$$\operatorname*{argmin}_{\mathbf{h}^j \geq 0} \|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 = \frac{\left[\mathbf{w}_j^\top \mathbf{M}_j\right]_+}{\|\mathbf{w}_j\|_2^2} = \frac{\left[(\mathbf{W}^\top \mathbf{M}_j)(j,:)\right]_+}{(\mathbf{W}^\top \mathbf{W})(j,j)}, \tag{1.4}$$

where $[\,\cdot\,]_+ = \max(\,\cdot\,, 0\,)$ is taken elementwise, $\mathbf{w}_j$ denotes the jth column of $\mathbf{W}$, and $\mathbf{M}_j$ is the residue matrix of $\mathbf{M}$ subtracting the approximation $\mathbf{W}\mathbf{H}$ except the rank-1 component $\mathbf{w}_j\mathbf{h}^j$, that is,

$$\mathbf{M}_j = \mathbf{M} - \sum_{i \neq j} \mathbf{w}_i \mathbf{h}^i = \mathbf{M} - \mathbf{W}\mathbf{H} + \mathbf{w}_j \mathbf{h}^j. \tag{1.5}$$

Note that such update is optimal, that is, no other update can performs better.

The algorithm using the update (1.5) is named HALS, and the algorithm updates the rows of $\mathbf{H}$ and the columns of $\mathbf{W}$ in a sequential way. By symmetry, the update of the jth column of $\mathbf{W}$ is

$$\operatorname*{argmin}_{\mathbf{w}_j \geq 0} \|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 = \frac{\left[\mathbf{M}_j \mathbf{h}^{j\top}\right]_+}{\|\mathbf{h}^j\|_2^2} = \frac{\left[(\mathbf{M}_j \mathbf{H}^\top)(:,j)\right]_+}{(\mathbf{H}\mathbf{H}^\top)(j,j)}. \tag{1.6}$$

That is, by replacing line 5 and line 6 in Algorithm 1 by Equation (1.6) and Equation (1.4) respectively, we obtain the HALS algorithm.

HALS has been improved in several ways.

- **Acceleration by using computed components** It is good to update the rows of $\mathbf{H}$ several times before updating $\mathbf{W}$ (and similarly for the columns of $\mathbf{W}$) as the computation of $\mathbf{W}^\top\mathbf{W}$ (which stores $\|\mathbf{w}_j\|_2^2$) and $\mathbf{W}^\top\mathbf{M}$ (which stores $\mathbf{w}_j^\top\mathbf{M}$ as the jth column) can be reused. Reusing computed terms reduce the computational load and allows for a significant acceleration of HALS [46]. This variant is referred to as accelerated HALS (AHALS). In §7 , we discuss an acceleration framework that speeds up AHALS (and also other NMF algorithms) significantly.

- **Stability improvement on preventing zero vector** It is possible that the HALS updates generate a zero vector [61], which may not be desirable if we do not want the rank of $\mathbf{W}$ to decrease. For example, the update (1.4) will give a zero vector if ${\mathbf{h}^j}^\top \in \text{Null}(\mathbf{M}_j)$, or $\mathbf{M}_j{\mathbf{h}^j}^\top \leq 0$. A numerical solution to prevent zero vector is to replace $[\,\cdot\,]_+$ as $\max(\,\cdot\,,\epsilon\,)$ with $\epsilon > 0$ as a small constant.

We will also improve (A)-HALS further by extrapolation; see §7 .

1.4.4 Solution space of NMF

Given a matrix $\mathbf{M}$, a factorization rank r, we now discuss the uniqueness of the solution of NMF. Let $(\mathbf{W}^\#, \mathbf{H}^\#)$ be an exact NMF solution of $\mathbf{M}$ with factorization rank r, i.e., $\mathbf{M} = \mathbf{W}^\#\mathbf{H}^\#$, where $\mathbf{W}^\# \in \mathbb{R}_+^{m\times r}$, $\mathbf{H}^\# \in \mathbb{R}_+^{r\times n}$, and $\text{rank}\,(\mathbf{W}^\#) = r$. We call the solution $(\mathbf{W}^\#, \mathbf{H}^\#)$ *essentially unique* if for any other (exact) NMF solution of the same size, denoted as $(\mathbf{W}^*, \mathbf{H}^*)$ with $\mathbf{W}^* \in \mathbb{R}_+^{m\times r}$ and $\mathbf{H}^* \in \mathbb{R}_+^{r\times n}$, there exist a permutation and a positive diagonal scaling matrix such that

$$\mathbf{W}^* = \mathbf{W}^\#\mathbf{\Lambda}\mathbf{\Pi}_r, \quad \mathbf{H}^* = \mathbf{\Pi}_r^{-1}\mathbf{\Lambda}^{-1}\mathbf{H}^\#, \tag{1.7}$$

where $\mathbf{\Pi}_r$ is an order-r permutation matrix and $\mathbf{\Lambda}$ is a diagonal matrix with positive diagonal. This equation implies all NMF solutions intrinsically contain two ambiguities: permutation and scaling, or sometimes called rotation and norm indeterminacy. This is illustrated as follows, for any $\lambda_j > 0$ and a permutation $\pi : [r] \to [r]$, then

$$\mathbf{W}^*\mathbf{H}^* = \sum_{j=1}^r \mathbf{W}^*(\,:, j)\mathbf{H}^*(j, :\,) = \sum_{j=1}^r \underbrace{\lambda_j\mathbf{W}^*(\,:, \pi(j))}_{\mathbf{W}^\#(:,j)} \underbrace{\frac{1}{\lambda_j}\mathbf{H}^*(\pi(j), :\,)}_{\mathbf{H}^\#(j,:)} = \mathbf{W}^\#\mathbf{H}^\#. \tag{1.8}$$

Equation (1.8) means that the solution space of NMF is unbounded, and if a matrix $\mathbf{M}$ has an NMF solution, there are infinitely many other solutions. In optimization terms, problem (1.1) has many global minima.

The issue of the identifiability of NMF is to ask the question: "what conditions on the NMF model will guarantee that, when solving such NMF problem, the solution is essentially unique?" In §2.1.5 , we give an identifiability theorem of a particular NMF model, where the main strategy in the proof is to make use of Equation (1.7) to derive a contradiction.

1.4.5 Geometry of NMF

We now give a geometric interpretation of NMF. Because of the nonnegativity, NMF is describing a cone sitting in the nonnegative orthant in $\mathbb{R}^m$, where the columns of $\mathbf{M}$, denoted as $\mathbf{m}_j$, are a collection of points in this space; see Fig.1.3 for an illustration. Such “data cone” (the cone built by the collections of the blue rays in Fig.1.3) is contained inside a polyhedral cone with extreme rays as the columns of $\mathbf{W}$ (the cone in red in Fig.1.3), which in turns is contained inside the nonnegative orthant (the green cone in Fig.1.3). Mathematically, $\text{cone}(\mathbf{M}) \subseteq \text{cone}(\mathbf{W}) \subseteq \mathbb{R}^m_+$. Under such geometric interpretation, NMF is a *Nested Polytope Problem* [45]: given a set of (nonnegative) points, find a polyhedral cone $\mathcal{C}$ in between two polyhedral cones, the first one is the data cone, and the second one is the nonnegative orthant. Here, the base of $\mathcal{C}$ is a polyhedral cone with r extreme rays, where r is exactly the factorization rank of $\mathbf{M}$.

1.4.6 NMF is NP-hard

The aforementioned NMF problems (the exact-NMF, the approximate-NMF, and the nonconvex program (1.1)) are all NP-hard, where the NP-hardness comes from the nonnegativity constraint. To be precise, here NP-hardness means that there does not exist an algorithm that solves NMF in polynomial time in the parameters (m, n, r). The NP-hard result was first shown in [59] implicitly, and later the result was shown explicitly in the seminal work of Vavasis [106]. Recently Yaroslav Shitov gave a short proof that NMF remains NP-hard when restricted to Boolean matrices [94]. We refer to the works [96, 95] by Shitov for more recent discussions on this topic.

Many new NMF models are proposed with additional assumptions to form a problem that is “less hard” to solve, is “easier to computer” or “has unique solution”. The additional constraints in these models shrink the solution space of the problem, and possibly also lead to identifiability. One of such NMF models that achieved great success is the *Separable NMF*.

1.5 Separable NMF

Separable NMF (SNMF) adds on top of NMF an extra condition on the matrix $\mathbf{M}$: the columns of $\mathbf{W}$ are copies of certain columns of $\mathbf{M}$. Mathematically,

$$\text{SNMF} : \mathbf{M} = \underbrace{\mathbf{M}(:,\mathcal{K})}_{\mathbf{W}} \underbrace{[\mathbf{I}_r \ \mathbf{H}']\mathbf{\Pi}_n}_{\mathbf{H}}, \tag{1.9}$$

where $\mathbf{M}(:,\mathcal{K})$ is the submatrix of $\mathbf{M}$ with columns labeled by the set $\mathcal{K}$, which contains r indices, $\mathbf{H}' \in \mathbb{R}^{r\times(n-r)}_+$ and $\mathbf{\Pi}_n$ is a n-by-n permutation matrix. Without loss of generality, we assume no permutation so $\mathcal{K} = [r]$, where $[r] := \{1, 2, \dots, r\}$. In this case SNMF (1.9) gives $\mathbf{W} = \mathbf{M}(:,\mathcal{K})\mathbf{\Lambda}^{-1} = \mathbf{M}(:,[r])\mathbf{\Lambda}^{-1}$, where $\mathbf{\Lambda}^{-1} = \text{Diag}(d_{jj}^{-1})$ is a positive diagonal matrix. We have $\mathbf{w}_j = d_{jj}^{-1}\mathbf{m}_j$ for all $j \in [r]$, meaning that the jth column of $\mathbf{W}$ is a scaled version of the jth column of $\mathbf{M}$. This means that the columns of $\mathbf{W}$ are “copies” of the data.

For the remaining data $\mathbf{M}(:,r+1:n)$, all columns are represented by these r columns in $\mathbf{W}$ (which are the first r columns of $\mathbf{M}$), so $\mathbf{M}(:,r+1:n) = \mathbf{M}(:,1:r)\mathbf{\Lambda}^{-1}\mathbf{H}'$, we can see that SNMF is a “self-dictionary” model: the basis that explains the data is contained within the data.

Geometry of SNMF The “self-expressiveness” in SNMF means that the polyhedral cone described by $\mathbf{W}$, which encapsulates all the data points $\mathbf{M}$, has r extreme rays passing through r points in the data; see Fig.1.3 for an illustration. Furthermore, on top of SNMF, if the columns of $\mathbf{H}$, denoted as $\mathbf{h}_j$, fulfill the condition $\mathbf{h}_j^\top \mathbf{1}_r = 1$, that is, sum to 1, then the aforementioned conical hull geometry reduces to a convex hull geometry in the $(m-1)$-dimensional space. In the example of Fig.1.3, such convex hull is indicated as the red triangle.

The state of the art SNMF problem can be solved in polynomial time and it is identifiable, i.e., SNMF recovers the ground truth matrices; provided that the data is generated in the form of (1.9) and that we know the correct factorization rank r, which can also be estimated. The main task in SNMF is to identify the set $\mathcal{K}$ in Equation (1.9), by which there are many (polynomial-time) algorithms available to solve this task. An example is the Successive Projection Algorithm (`SPA`) [7], which is essentially a modified Gram-Schmidt procedure, to be described below in §1.5.1 . Furthermore, `SPA` is provably robust to bounded additive noise [47]. A modified `SPA` called `SNPA` has been proposed to deal with rank-deficient problems [43]. `SPA` is one of the best algorithm for solving SNMF [47] because: (i) it is far more practical than other algorithms in terms of computational cost, (ii) it has no parameter to tune, (iii) it works even when the data is not separable. For other SNMF algorithms, see [47] and the references therein.

Relaxing the separability condition The success of SNMF algorithms relies on the separability assumption in the SNMF model (1.9). Relaxing this condition motives the study of a more general NMF model called *Minimum volume NMF* (minvol NMF), which is the focus of the next chapter.

1.5.1 Successive Projection Algorithm

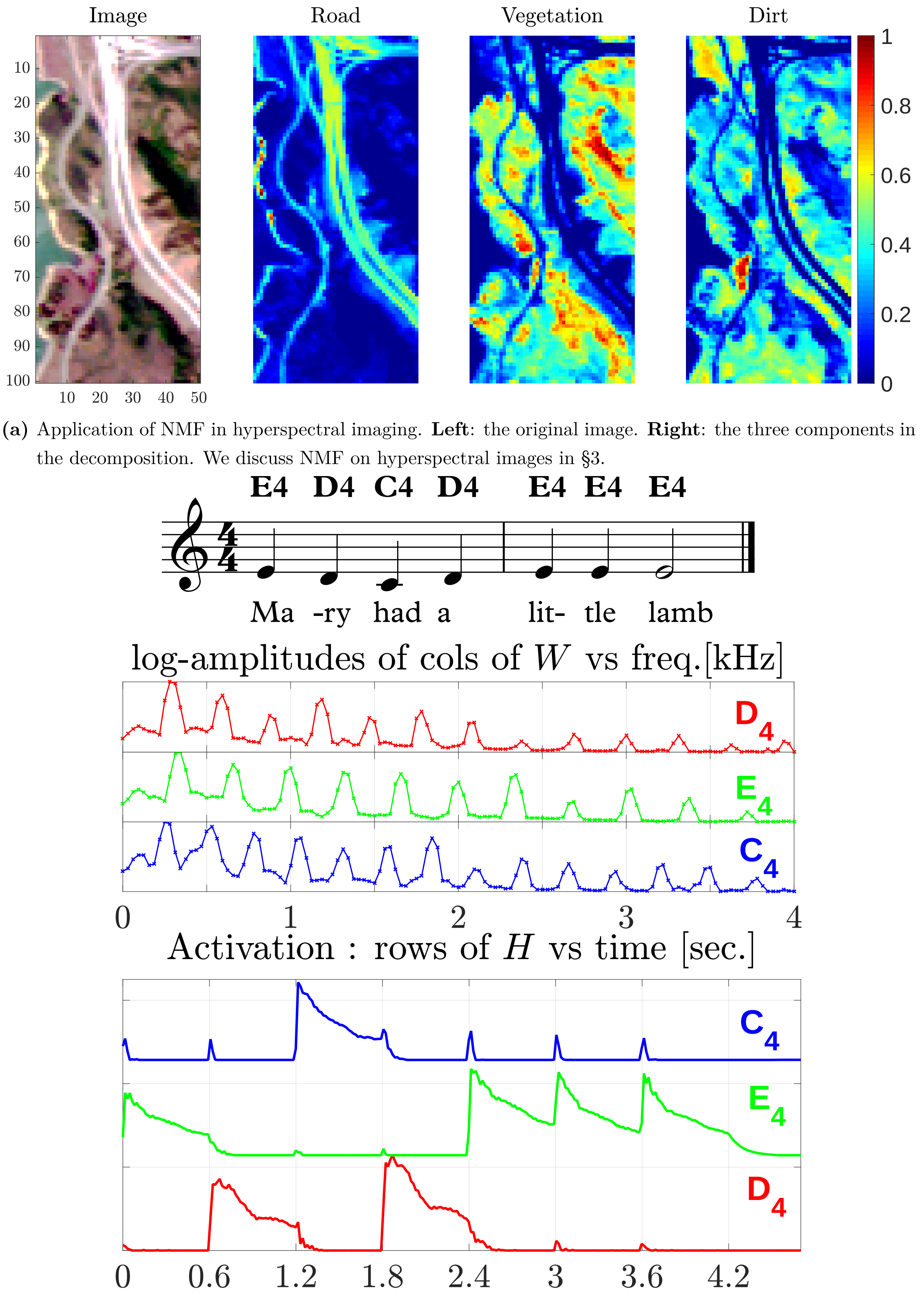
Before we move to minvol NMF, we briefly describe SPA; see Algorithm 2. This thesis makes heavy uses of `SPA` in various experiments, hence it is better to have a basic understanding of it. For a detail discussion on `SPA` such as the convergence, computational complexity, robustness to additive noise, we refer to the works [47, 43] and the references therein.

In a nutshell, `SPA` is a greedy algorithm able to identify the set $\mathcal{K}$ in Equation (1.9). Given an input matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ and a factorization rank r, `SPA` first picks the column in $\mathbf{M}$ with the largest L_2-norm as the first index for the set $\mathcal{K}$. Then, it removes the contribution of such column, by projecting the remaining data columns of $\mathbf{M}$ to the orthogonal complement of the extracted column. Such extraction-projection procedure is repeated r times, and therefore giving r column indices for the set $\mathcal{K}$.

Algorithm 2 `SPA`: Successive Projection Algorithm [7]

```
Input: $\mathbf{M}$ (a a matrix suppose to be generated as the SNMF model (1.9)), $r$ (number of indices to be extracted)
Output: a set $\mathcal{K}$ with $r$ indices
Initialization: $\mathcal{K} = [\cdot]$, $\mathbf{R} = \mathbf{M}$.
for $k = 1, \dots r$ do
    $p = \underset{j}{\operatorname{argmax}} \|\mathbf{R}(:,j)\|_2$                % Index selection
    $\mathcal{K} = \mathcal{K} \cup \{p\}$
    $\mathbf{R} = \left(\mathbf{I} - \frac{\mathbf{R}(:,p)\mathbf{R}(:,p)^\top}{\|\mathbf{R}(:,p)\|_2^2}\right)\mathbf{R}$                % Projection
end for
```

Lastly we briefly mention how to use `SPA` to solve SNMF. After obtaining the set $\mathcal{K}$, we put $\mathbf{W} = \mathbf{M}(:,\mathcal{K})$, then we can find $\mathbf{H}$ by solving a NNLS, i.e., Problem (1.3).



(a) Application of NMF in hyperspectral imaging. **Left**: the original image. **Right**: the three components in the decomposition. We discuss NMF on hyperspectral images in §3.

(b) Application of NMF in audio source separation. NMF can be used to perform blind source separation on single channel audio recording data. **Top**: the music score "Mary had a little lamb". **Bottom**: the three components in the decomposition. We discuss NMF on audio blind source separation in §4.

Fig. 1.1. Applications of NMF: in hyperspectral imaging and audio source separation.

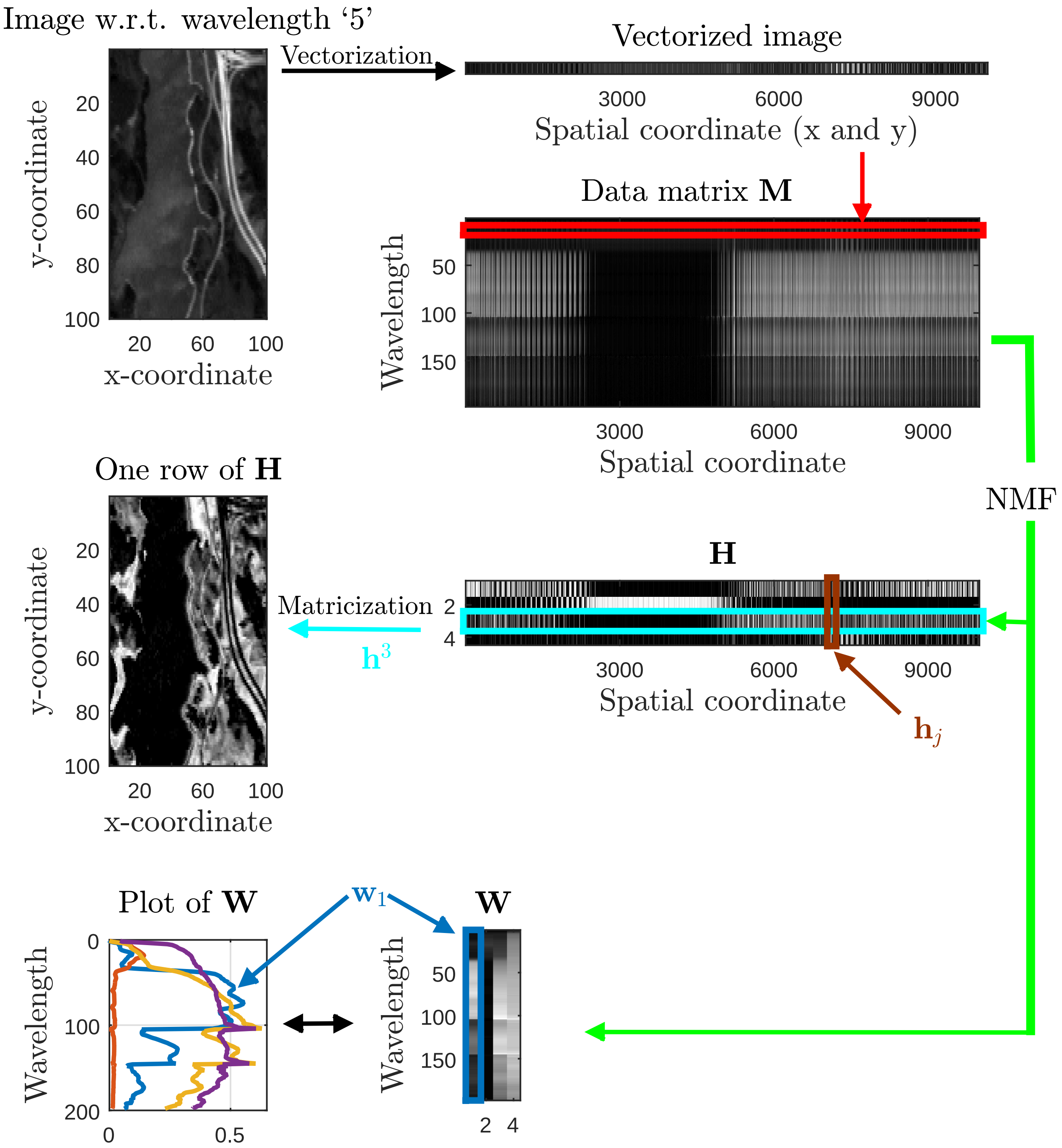


Fig. 1.2. A pictorial description on how NMF is used in hyperspectral imaging application. The data matrix **M** consists of two modalities: wavelength (row dimension) and spatial coordinate (column dimension). The physical meaning of the NMF factor matrices **W** and **H** is then associated to the row and column modalities of the data matrix.

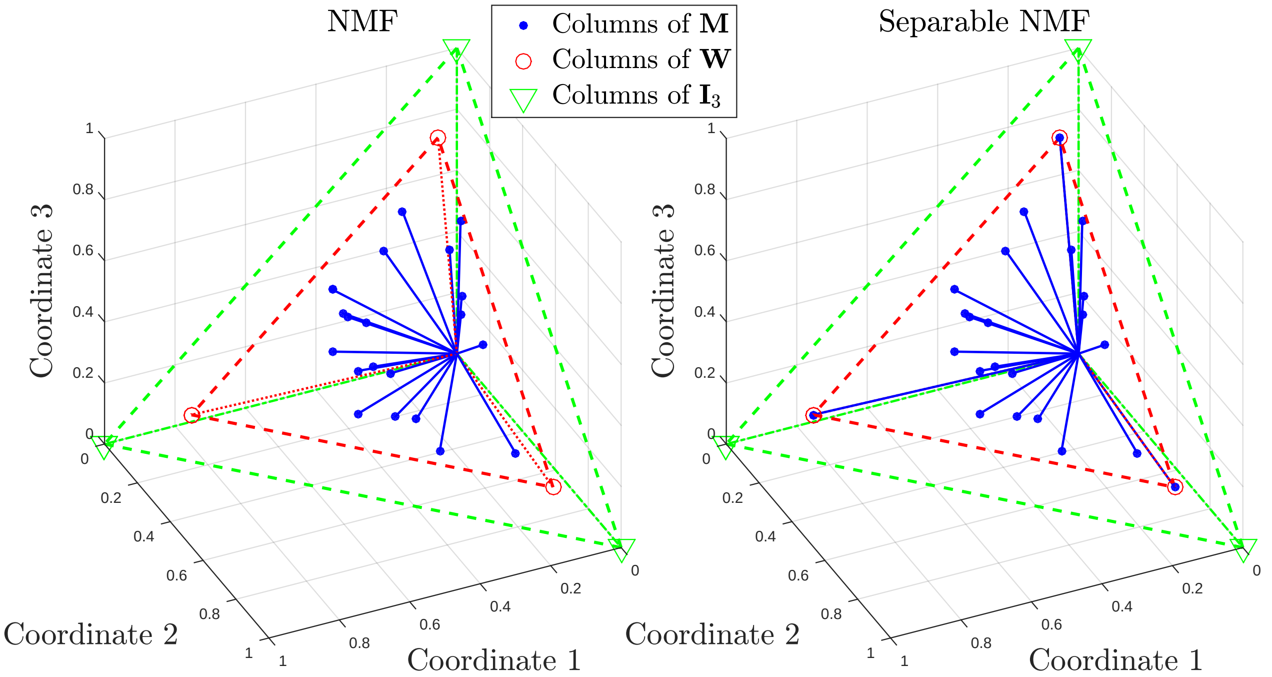


Fig. 1.3. Example of NMF and SNMF. Here $r = m = 3$, $n = 20$. The blue rays are data points $\mathbf{M}$, the red rays are the columns of $\mathbf{W}$ and the green rays are the standard basis vectors in $\mathbb{R}^3$. In both cases, blue cone $\subseteq$ red cone $\subseteq$ green cone.

2 NMF with minimum volume: minvol NMF

> The more you know the more you realize you don't know.
>
> *Aristotle*

In this chapter, we discuss a specific class of NMF model named minvol NMF.

Chapter organization We first talk about a generalization of the Separable NMF called minimum-volume NMF (minvol NMF) in §2.1 , where we discuss about the modeling issues and provide an identifiability theorem on minvol NMF. In §2.2, we briefly discuss how to solve minvol NMF. Finally, we conclude this chapter in §2.3, where we list some open problems related to minvol NMF.

Highlights of contributions Contributions start at §2.1 when we introduce minvol NMF with the nuclear norm used as a regularization term. For the minvol NMF with spectral function regularizers such as determinant and log-determinant, they they have already been investigated in the literature, but we give further and new discussions. We discuss their properties and how to tune parameters, where in the experiments we show that the minvol NMF with log-determinant regularizer with our parameter tuning strategy performs better than existing approach. Then in §2.1.5 we give an new identifiability theorem on a particular minvol NMF model, where the proof is modified from existing techniques. We then compare our minvol NMF model to other similar models and argue that our model better suited for applications. In §2.2 , we derive algorithm for solving minvol NMF problems, by improving some existing techniques. We also propose a new heuristic to solve minvol NMF with nuclear norm regularization.

2.1 Minimum-volume NMF

In this section, we focus on minimum-volume NMF (minvolNMF). We first present the formulation of minvol NMF in §2.1.1 , then we focus on the regularizer in §2.1.2 and §2.1.3 . Finally we discuss a special minvol NMF model that is provably identifiable in §2.1.5 , i.e., solving minvol NMF recovers the ground truth matrices that generate the data, under some conditions.

2.1.1 Formulation of minvol NMF

Minvol NMF is the following linear algebra problem: given $\mathbf{M}$, find the pair $(\mathbf{W}, \mathbf{H})$ such that $\mathbf{M} \approxeq \mathbf{WH}$ subject to the condition that the "volume" of $\mathbf{W}$ is minimized. Refer to the NMF example in Fig.1.3, it means that we try to find the smallest red cone to encapsulate all the blue points. Note that in the SNMF example, the red cone is already the smallest cone that captures all the blue points.

Minvol NMF can be cast as the following nonconvex optimization problems:

$$\text{(H model)} : \underset{\mathbf{W}\geq 0, \mathbf{H}\geq 0}{\text{minimize}} \ \frac{1}{2}\|\mathbf{M} - \mathbf{WH}\|_F^2 + \lambda \mathcal{V}(\mathbf{W}) \ \text{ subject to } \ \mathbf{H}^\top \mathbf{1}_r \leq \mathbf{1}_n, \tag{2.1a}$$

$$\text{(W model)} : \underset{\mathbf{W}\geq 0, \mathbf{H}\geq 0}{\text{minimize}} \ \frac{1}{2}\|\mathbf{M} - \mathbf{WH}\|_F^2 + \lambda \mathcal{V}(\mathbf{W}) \ \text{ subject to } \ \mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r, \tag{2.1b}$$

where $\mathcal{V} : \mathbb{R}^{m \times r} \to \mathbb{R}$ is a *"volume" function* that quantifies the volume of $\mathbf{W}$, the constant $\lambda \geq 0$ is the regularization parameter, and $\mathbf{1}_r$ denotes vector of ones in $\mathbb{R}^r$. The difference between the two models (2.1a) and (2.1b) is the normalization. The constraint $\mathbf{H}^\top \mathbf{1}_r \leq \mathbf{1}_n$ is the normalization of columns of $\mathbf{H}$, i.e., $\mathbf{h}_j^\top \mathbf{1}_r \leq 1$ for each columns $\mathbf{h}_j$; and the constraint $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$ is the normalization of column of $\mathbf{W}$, i.e., ${\mathbf{w}_j}^\top \mathbf{1}_m = 1$ for each columns $\mathbf{w}_j$.

Both normalizations are applied to the columns of $\mathbf{H}$ or the columns of $\mathbf{W}$, which is to regulate the behavior of these column vector, refer back to Table 1.1 for the physical meaning of the column vectors. The normalization serves multiple purposes.

- First, it removes scaling ambiguity (see Equation (1.8)).
  - The constraint $\mathbf{H}^\top \mathbf{1}_r \leq \mathbf{1}_n$ implies $\mathbf{H} \leq \mathbf{1}_{r \times n}$ since $\mathbf{H} \geq 0$. It means that $\mathbf{W}$ cannot go to zero in the optimization process, otherwise it will cause the matrix product $\mathbf{WH}$ to also go zero.
  - For the constraint $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$, it implies $\mathbf{W} \leq \mathbf{1}_{m \times r}$ since $\mathbf{W} \geq 0$.
- For the constraint $\mathbf{H}^\top \mathbf{1}_r \leq \mathbf{1}_n$, it is application driven. The constraint means that $\mathbf{h}_j^\top \mathbf{1}_r \leq 1$ for each columns $\mathbf{h}_j$, i.e., the entries of $\mathbf{h}_j$ sum-to-at-most-1. In many applications, this corresponds to conservation law of the physical object NMF is modeling. In §3 , we will discuss using NMF on hyperspectral imaging applications, where the model makes uses of the constraint $\mathbf{H}^\top \mathbf{1}_r \leq \mathbf{1}_n$.
- For the constraint $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$, it is algorithmic driven. The constraint means that all columns of $\mathbf{W}$ have the same L_1-norm, which prevents the condition number of $\mathbf{W}^\top \mathbf{W}$ to be too large. A good condition number of $\mathbf{W}^\top \mathbf{W}$ facilitates the update of both $\mathbf{W}$ and $\mathbf{H}$ in the algorithm proposed to solve the minvol NMF. In short, the constraint makes the algorithm numerically more stable. We will discuss this issue in details in §2.2 .

Notice that the normalization constraint in minvol NMF problem (2.1a) is defined using the inequality instead of equality:

$$\mathbf{H}(:, j) \in \Delta^r := \left\{ \mathbf{x} \in \mathbb{R}_+ \mid \sum_{i=1}^{r} x_i \leq 1. \right\}. \tag{2.2}$$

In this thesis, we simply call (2.2) as the *Simplex constraint.*

Other motivations of minvol NMF Apart from the aforementioned fact that minvol NMF generalizes SNMF, there are two more factors that motivate the study of minvol NMF.

- The first one is theory-driven. The study of minvol NMF is closely related to the identifiability of NMF. As discussed in §1 on the geometric interpretation of NMF and SNMF, geometrically minvol NMF finds the smallest cone that captures the data points, in which the vertices of the cone($\mathbf{W}$) recover the ground truth generating vertices of the data points. See the discussions on geometry in §1.4 and §1.5 on Fig.1.3. Furthermore, in §2.1.5 , we will give an identifiability result for the minvol NMF model (2.1b).
- The second reason is that minvol NMF is application-driven. For example, in the remote sensing community, the minvol criterion is called "Craig's belief"[25] in some literature [81]. In §3 and §4 , we will demonstrate the effectiveness of minvol NMF models on two real-world applications, namely the hyperspectral imaging and audio source separation.

2.1.2 Volume regularizers as singular value regularizer

The key ingredient in minvol NMF is the function $\mathcal{V}(\mathbf{W})$ that measures the "volume" of the matrix $\mathbf{W}$. Recall that for a square matrix $\mathbf{W}$ that has full rank, the (signed) volume of the parallelepiped generated by the columns of $\mathbf{W}$ is given by $\det(\mathbf{W})$. In real-world applications, in general $\mathbf{W}$ is rectangular so determinant function cannot be used, unless a linear transform such as Principal component analysis (PCA) is applied on $\mathbf{W}$ to make it square [84]. However, such dimension reduction destroys the nonnegativity of $\mathbf{W}$.

To keep the nonnegativity, in this thesis, we focus on the Gram matrix $\mathbf{W}^\top\mathbf{W}$. In fact, by considering the Gram matrix, we immediately have our first volume function:

Lemma 2.1.1. *For $\mathbf{W} \in \mathbb{R}^{m\times r}$, $m \geq r$ that is full rank, then*

$$\sqrt{\det(\mathbf{W}^\top\mathbf{W})} \tag{2.3}$$

is the volume of the convex hull of the columns of $\mathbf{W}$ and the origin, i.e., the volume of conv$([\mathbf{0}, \mathbf{W}])$, in the column space of $\mathbf{W}$, subject to a proportion constant.

Sketch of the proof Below we give the proof of the lemma. For the proportion constant, see [101].

First, we recall the fact that determinant is the volume of parallelepiped in any dimensions. This can be shown using Gram-Schmidt orthogonalization. Given a set of vectors $\{\mathbf{u}_1, \dots, \mathbf{u}_n\}$, Gram-Schmidt orthogonalization on this set of vectors gives

$$\begin{aligned}
\mathbf{v}_1 &= \mathbf{u}_1 \\
\mathbf{v}_2 &= c_{12}\mathbf{u}_1 + \mathbf{u}_2^\perp \\
\mathbf{v}_3 &= c_{13}\mathbf{u}_1 + c_{23}\mathbf{u}_2^\perp + \mathbf{u}_3^\perp \\
&\vdots
\end{aligned}$$

where $\mathbf{u}_2^\perp$ is orthogonal to $\mathbf{v}_1$; $\mathbf{u}_3^\perp$ is orthogonal to span$(\mathbf{v}_1, \mathbf{v}_2)$, and so on. Then

$$\begin{aligned}
|\det(\mathbf{v}_1, \dots, \mathbf{v}_n)| &= \det(\mathbf{u}_1, c_{12}\mathbf{u}_1 + \mathbf{u}_2^\perp, c_{13}\mathbf{u}_1 + c_{23}\mathbf{u}_2^\perp + \mathbf{u}_3^\perp, \dots) \\
&= \det(\mathbf{u}_1, \mathbf{u}_2^\perp, \mathbf{u}_3^\perp, \dots, \mathbf{u}_n^\perp) \\
&= \text{volume}(\mathbf{u}_1, \mathbf{u}_2^\perp, \mathbf{u}_3^\perp, \dots, \mathbf{u}_n^\perp),
\end{aligned} \tag{2.4}$$

where the first equality comes from the Gram-Schmidt orthogonalization, the second equality comes from the property of determinant, and the last equality is due to the fact that the vectors $\mathbf{u}_1, \mathbf{u}_2^\perp, \dots, \mathbf{u}_n^\perp$ are orthogonal to each other. Such equation means that the volume of the parallelepiped generated by a set of orthogonal vectors can be computed using determinant on the orthogonal basis vectors of that set of vectors.

Now, consider the volume of the parallelepiped of $[0, \mathbf{W}]$. It is related to the column space of $\mathbf{W}$, in which we can use SVD to get the orthogonal basis. As $\mathbf{W}$ has rank r, let the rank-r truncated SVD of $\mathbf{W}$ be $\mathbf{U\Sigma V}^\top$. As $\mathbf{U}$ is an orthogonal basis of $\mathbf{W}$, the orthogonal vectors representing $\mathbf{W}$ in the basis $\mathbf{U}$ are the vectors $\sigma_1\mathbf{v}_1, \dots, \sigma_r\mathbf{v}_r$, where $\mathbf{v}_i$ is the ith column of $\mathbf{V}$ and σ_i is the ith largest singular value. Using the determinant volume formula (2.4), the volume of the conv$([\mathbf{0}, \mathbf{W}])$ in the column space of $\mathbf{W}$ is proportional to

$$|\det(\sigma_1\mathbf{v}_1, \dots, \sigma_r\mathbf{v}_r)| = |\det(\Sigma\mathbf{V}^\top)| = |\det(\Sigma)| = \sqrt{\det(\mathbf{W}^\top\mathbf{W})}.$$

The sketch of the proof is now completed.

It is crucial to note that the notion of "volume" is the volume of conv($[\mathbf{0}, \mathbf{W}]$) in the column space of $\mathbf{W}$. Also, note that $\mathbf{W}$ has to be full rank, a rank-deficient $\mathbf{W}$ corresponds to a "flat" cone with volume zero.

Note that if $\mathbf{H}$ is normalized as in minvol NMF problem (2.1a), the conical hull geometry described in §1.4 , §1.5 and Fig.1.3 is changed to convex hull geometry, as shown as the triangles in Fig.1.3.

Table 2.1 shows some examples of possible volume regularizers. Intuitively, as the "volume" of a matrix is related to its singular values, we can see that these examples are all functions of singular values, and therefore, we can interpret minvol NMF as a constrained matrix recovery problem of the form

$$\min_{\mathbf{W} \geq 0, \mathbf{H} \geq 0} \quad g \circ \boldsymbol{\sigma}(\mathbf{W}) + \frac{1}{2}\|\mathbf{W}\mathbf{H} - \mathbf{M}\|_F^2, \tag{2.5}$$

where $\mathcal{V}(\mathbf{W}) = g \circ \boldsymbol{\sigma}(\mathbf{W})$ is a composite function, the symbol $\circ$ denotes the composition operator, the function $g : \mathbb{R}^r \to \mathbb{R}$ and the symbol $\boldsymbol{\sigma}(\mathbf{W})$ denotes the vector of singular values of $\mathbf{W}$. Note that in (2.5), we do not show explicitly the normalization constraint.

Table 2.1: Some examples of $\mathcal{V}(\mathbf{W})$.

Name	Definition	In $g \circ \boldsymbol{\sigma}(\mathbf{W})$
Determinant (det) [6, 4]	$\mathcal{V}_{\text{det}}(\mathbf{W}) = \det\left(\mathbf{W}^\top \mathbf{W}\right)$	$\prod_{i=1}^r \sigma_i^2$
log-determinant (logdet) [6, 4]	$\mathcal{V}_{\text{logdet}}(\mathbf{W}) = \text{logdet}\left(\mathbf{W}^\top \mathbf{W} + \delta \mathbf{I}_r\right)$	$\sum_{i=1}^r \log\left(\sigma_i^2 + \delta\right)$
Frobenius norm squared	$\mathcal{V}_{\text{F}}(\mathbf{W}) = \|\mathbf{W}\|_F^2$	$\sum_{i=1}^r \sigma_i^2$
Nuclear norm [6]	$\mathcal{V}_*(\mathbf{W}) = \|\mathbf{W}\|_*$	$\sum_{i=1}^r \sigma_i$
Smooth Schatten-p norm [33]	$\mathcal{V}_{p,\delta}(\mathbf{W}) = \text{Tr}\left(\mathbf{W}^\top \mathbf{W} + \delta \mathbf{I}_r\right)^{\frac{p}{2}}$	$\sum_{i=1}^r (\sigma_i^2 + \delta)^{\frac{p}{2}}$

Discussion on the volume regularizers Before we move to the next subsection, we give some specific remarks on the volume regularizers listed in Table 2.1.

- **On $\mathcal{V}_{\text{det}}$.** The use of $\mathcal{V}_{\text{det}}$ is motivated by removing the square root in the expression (2.3) for the ease of computing it, explained as follows. In fact $\mathcal{V}_{\text{det}}(\mathbf{w}_j)$ is nonconvex in $\mathbf{W}$ but convex in one column of $\mathbf{W}$: the function $\mathcal{V}_{\text{det}}(\mathbf{w}_j)$ is a quadratic form $\gamma_j \mathbf{w}_j \mathbf{Q}_i \mathbf{w}_j$ where $\gamma_j \geq 0$ is a constant and $\mathbf{Q}_i$ is a symmetric matrix that is positive definite if $\mathbf{W}$ is full rank. This is a standard result in linear algebra. For completeness, we show the derivation here. Let $\mathbf{W}_{-j}$ be

$\mathbf{W}$ without the column $\mathbf{w}_j$, then $\mathbf{W} = [\mathbf{w}_j, \mathbf{W}_{-j}]\mathbf{\Pi}$, and $\mathcal{V}_{\text{det}}(\mathbf{W}) = \det(\mathbf{W}^\top \mathbf{W})$ becomes

$$\begin{aligned}
\det(\mathbf{W}^\top\mathbf{W}) &= \det\left(\mathbf{\Pi}^\top \begin{bmatrix} \mathbf{w}_j^\top \\ \\ \mathbf{W}_{-j}^\top \end{bmatrix} \begin{bmatrix} \mathbf{w}_j \ \mathbf{W}_{-j} \end{bmatrix} \mathbf{\Pi}\right) \\
&= \det\left(\begin{bmatrix} \mathbf{w}_j^\top \mathbf{w}_j & \mathbf{w}_j^\top \mathbf{W}_{-j} \\ \\ \mathbf{W}_{-j}^\top \mathbf{W}_{-j} & \mathbf{W}_{-j}^\top \mathbf{W}_{-j} \end{bmatrix} \underbrace{\mathbf{\Pi}\mathbf{\Pi}^\top}_{=\mathbf{I}}\right) \\
&= \underbrace{\det\left(\mathbf{W}_{-j}^\top \mathbf{W}_{-j}\right)}_{=\gamma_j} \det\left(\mathbf{w}_j^\top \mathbf{w}_j - \mathbf{w}_j^\top \mathbf{W}_{-j} (\mathbf{W}_{-j}^\top \mathbf{W}_{-j})^{-1} \mathbf{W}_{-j}^\top \mathbf{w}_j\right) \\
&= \gamma_j \det\left(\mathbf{w}_j^\top \left(\mathbf{I}_m - \mathbf{W}_{-j} (\mathbf{W}_{-j}^\top \mathbf{W}_{-j})^{-1} \mathbf{W}_{-j}^\top\right) \mathbf{w}_j\right) \\
&= \gamma_j \mathbf{w}_j^\top \underbrace{\left(\mathbf{I}_m - \mathbf{W}_{-j} (\mathbf{W}_{-j}^\top \mathbf{W}_{-j})^{-1} \mathbf{W}_{-j}^\top\right)}_{=\mathbf{Q}_j} \mathbf{w}_j := \mathcal{V}_{\text{det}}(\mathbf{w}_j),
\end{aligned} \tag{2.6}$$

where the third equality comes from the Schur complement. The computational cost of minimizing $\mathcal{V}_{\text{det}}$ comes from computing $\mathbf{Q}_j$. When m is large, the computational cost of minimizing minvol NMF with $\mathcal{V}_{\text{det}}$ becomes high. We will discuss this issue further in §2.2.2 .

It is important to note that the matrix $\mathbf{Q}_j$ is in fact the projector onto the complement of the column space of $\mathbf{W}_{-j}$, that is, mathematically $\mathbf{Q}_j$ is the projector on $\text{span}(\mathbf{W}_{-j})^\perp = \ker \mathbf{W}_{-j}^\top$. In this sense, $\mathcal{V}_{\text{det}}(\mathbf{w}_j)$ is proportional to the squared-distance between $\mathbf{w}_j$ and $\text{span}(\mathbf{W}_{-j})$. Knowing this fact, the function $\det(\mathbf{W}^\top\mathbf{W})$ can be expressed as a product of L_2 norm-squared as follows: given a vector $\mathbf{a}$, let $\mathbf{P}_{\mathbf{a}}^\perp = \mathbf{I} - \dfrac{\mathbf{a}\mathbf{a}^\top}{\|\mathbf{a}\|_2^2}$ be the projection onto the complement of the column space of $\mathbf{a}$, then based on Equation (2.4), we have

$$\mathcal{V}_{\text{det}}(\mathbf{w}_j) = \|\mathbf{w}_1\|_2^2 \cdot \left\|\mathbf{P}_1^\perp \mathbf{w}_2\right\|_2^2 \left\|\mathbf{P}_{1,2}^\perp \mathbf{w}_3\right\|_2^2 \cdots \left\|\mathbf{P}_{1,\dots,r-1}^\perp \mathbf{w}_r\right\|_2^2, \tag{2.7}$$

where

$$\begin{array}{rcll}
\mathbf{P}_1^\perp &=& \mathbf{P}_{\mathbf{a}_1}^\perp, & \mathbf{a}_1 = \mathbf{w}_1 \\
\mathbf{P}_{1,2}^\perp &=& \mathbf{P}_1^\perp \mathbf{P}_{\mathbf{a}_2}^\perp, & \mathbf{a}_2 = \mathbf{P}_1^\perp \mathbf{w}_2 \\
\mathbf{P}_{1,2,3}^\perp &=& \mathbf{P}_{1,2}^\perp \mathbf{P}_{\mathbf{a}_3}^\perp, & \mathbf{a}_3 = \mathbf{P}_{1,2}^\perp \mathbf{w}_3 \\
\vdots && \vdots & \\
\mathbf{P}_{1,\dots,r-1}^\perp &=& \mathbf{P}_{1,2,\dots,r-2}^\perp \mathbf{P}_{\mathbf{a}_{r-1}}^\perp, & \mathbf{a}_{r-1} = \mathbf{P}_{1,2,\dots,r-2}^\perp \mathbf{w}_{r-1},
\end{array}$$

see also [18, Lemma 3] .

- **On $\mathcal{V}_{\text{logdet}}$.** The use of $\mathcal{V}_{\text{logdet}}$ is motivated by the fact that the log operator can weight down dominant singular values as in $\mathcal{V}_{\text{det}}$. It is nonconvex in $\mathbf{W}$ but concave in $\mathbf{W}^\top\mathbf{W}$, which is to be discuss in Lemma 2.2.1. The $\delta\mathbf{I}$ term with $\delta \geq 0$ act as a lower bound of the function. The tuning of δ is important. When δ is small, it is possible that $\mathcal{V}_{\text{logdet}}(\mathbf{W})$ gives a negative value, and in particular, if $\delta = 0$, the function approaches to $-\infty$ as $\mathbf{W}$ approaches to a rank deficient matrix. We discuss the issue on choosing δ in §2.1.3 .

- **On $\mathcal{V}_{\mathbf{F}}$ and $\mathcal{V}_*$.** Both are convex in $\mathbf{W}$. The use of these functions are motivated by the fact that $\mathcal{V}_{\text{det}}$ and $\mathcal{V}_{\text{logdet}}$ are both increasing function in singular values.

- **On $\mathcal{V}_{p,\delta}$.** Here $p \geq 0, \delta \geq 0$. This function is convex in $\mathbf{W}$ if $p \geq 1$, and nonconvex if $p < 1$. This function generalizes $\mathcal{V}_{\text{logdet}}$, $\mathcal{V}_{\text{F}}$ and $\mathcal{V}_*$.

- **On the dispersion norm**. Some works (for example [90]) considered a function called dispersion norm $\mathcal{V}_{\text{d}}(\mathbf{W}) := \text{Tr}\left(\mathbf{W}^\top(\mathbf{I}_r - \dfrac{1}{r}\mathbf{1}_r\mathbf{1}_r^\top)\mathbf{W}\right)$, which is closely related to the functions listed in Table 2.1. This convex function can be interpreted as a measure of the dispersion of the columns of $\mathbf{W}$ around their centroid [90], however it is unclear whether the dispersion norm is a function of singular values.

- **Relations between the $\mathcal{V}$s**. The volume regularizers in Table 2.1 are all increasing function in singular value of $\mathbf{W}$, showing that these functions are somehow related:
  - On $\mathcal{V}_{p,\delta}$ and $\mathcal{V}_{\text{F}}$: $\mathcal{V}_{2,0} = \mathcal{V}_{\text{F}}$.
  - On $\mathcal{V}_{p,\delta}$ and $\mathcal{V}_*$: $\mathcal{V}_{1,0} = \mathcal{V}_*$.
  - On $\mathcal{V}_{p,\delta}$ and $\mathcal{V}_{\text{logdet}}$. As illustrated in [87],

    $$\lim_{p\to 0} \frac{\mathcal{V}_{p,\delta}(\mathbf{W}) - r}{p} = \frac{1}{2}\mathcal{V}_{\text{logdet}}(\mathbf{W}).$$

  - As a side note, we also have

    $$\lim_{\substack{p\to 0\\ \delta\to 0}} \mathcal{V}_{p,\delta}(\mathbf{W}) = \lim_{\substack{p\to 0\\ \delta\to 0}} \sum_i (\sigma_i^2 + \delta)^{\frac{p}{2}} = \lim_{p\to 0} \sum_i (\sigma_i^2)^{\frac{p}{2}} = \lim_{p\to 0} \sum_i \sigma_i^p = \|\boldsymbol{\sigma}(\mathbf{W})\|_0 = \text{rank}\,(\mathbf{W}).$$

In the remaining parts in this chapter, we mainly focus on $\mathcal{V}_{\text{det}}, \mathcal{V}_{\text{logdet}}$ and $\mathcal{V}_*$.

2.1.3 More discussion on det and logdet volume: on the rank deficiency and parameter tuning

We now compare $\mathcal{V}_{\text{det}}$ and $\mathcal{V}_{\text{logdet}}$. In general we consider $\mathcal{V}_{\text{logdet}}$ a better volume regularizer than $\mathcal{V}_{\text{det}}$, due to the following three reasons.

- **det volume has high computational cost for the quadratic form** (2.6). In the works [4, 6], the approach used to solve solving minvol NMF with $\mathcal{V}_{\text{det}}$ is based on the quadratic form (2.6), which has a high computational cost if the dimension m is large. The high cost is mainly due to the computation of the matrix $\mathbf{Q}_i$ in (2.6). We will discuss this issue further in §2.2.2 .

- **logdet function is concave**. In terms of optimization, the function logdet$(\mathbf{X})$ is concave in $\mathbf{X}$ and leads to a nice majorizer; see Lemma (2.2.1).

- **logdet volume is a more balanced measure on the singular spectrum of the matrix $\mathbf{W}$**. Refer to the last column in Table 2.1, $\mathcal{V}_{\text{det}}$ is sensitive to the dominant singular values, while the log term in $\mathcal{V}_{\text{logdet}}$ allows to weight down dominant singular values, and makes it a more balanced measure on the singular spectrum of the matrix $\mathbf{W}$, provided that the δ parameter is properly chosen. It has been observed that $\mathcal{V}_{\text{logdet}}$ works better in practice in the hyperspectral imagining application [6], to be discussed in §3 .

- **logdet volume can be used in rank deficient situations**. Another advantage of $\mathcal{V}_{\text{logdet}}$ over $\mathcal{V}_{\text{det}}$ is that the former can detect rank deficiency. When $\mathbf{W}$ is rank deficient (i.e., $\text{rank}\,(\mathbf{W}) < r$), some singular values of $\mathbf{W}$ equal to zero so $\det(\mathbf{W}^\top\mathbf{W}) = 0$. Hence $\mathcal{V}_{\text{det}}$ cannot distinguish between different rank deficient solutions. If one adds a perturbation and consider $\mathcal{V}_{\text{det}} = \det(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_r)$, then δ has to be tuned extremely precise as the function $\det(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_r) = \prod_i(\sigma_i^2 + \delta)$ is very sensitive to δ. Meanwhile, $\mathcal{V}_{\text{logdet}}(\mathbf{W}) = \sum_i \log\left(\sigma_i^2 + \delta\right)$ so if $\mathbf{W}$ has one (or more) zero singular value, this measure still makes sense: among two rank-deficient solutions belonging to the same low-dimensional subspace, minimizing $\mathcal{V}_{\text{logdet}}$ will favor a solution whose convex hull has a smaller volume within that subspace since decreasing the non-zero singular values of $(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I})$ decreases $\mathcal{V}_{\text{logdet}}$. Mathematically, let $\mathbf{W} \in \mathbb{R}^{m\times r}$ belong to a k-dimensional subspace with $k < r$ so that $\mathbf{W} = \mathbf{US}$ where $\mathbf{U} \in \mathbb{R}^{m\times k}$ is an orthogonal basis of that subspace and $\mathbf{S} \in \mathbb{R}^{k\times r}$ are the coordinates of the columns of $\mathbf{W}$ in that subspace. Then $\mathcal{V}_{\text{logdet}}(\mathbf{W}) = \text{logdet}(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I})$ becomes

$$\mathcal{V}_{\text{logdet}}(\mathbf{W}) = \log\det(\mathbf{S}^\top\mathbf{S} + \delta\mathbf{I}) = \log\prod_{i=1}^{k}\Big(\sigma_i^2(\mathbf{S}) + \delta\Big)\delta^{r-k} = \sum_{i=1}^{k}\log\Big(\sigma_i^2(\mathbf{S}) + \delta\Big) + (r-k)\log\delta\ .$$

The minvol criterion $\mathcal{V}_{\text{logdet}}$ with $\delta > 0$ is therefore meaningful even when $\mathbf{W}$ does not have rank r. This has been observed in [71] on rank deficient cases. An application is that this automatically performs model order selection, by setting redundant components to zero. This has been observed in audio source separation where the factorization rank r is overestimated; for example, see [72], and §4.2 . We will go back to this issue in §4.2 where we discuss the use of minvol NMF to solve audio source separation problems.

As $\mathcal{V}_{\text{logdet}}$ is a better volume function, we discuss further on the parameter tuning of $\mathcal{V}_{\text{logdet}}$.

The choice of δ in $\mathcal{V}_{\text{logdet}}(\mathbf{W})$ The function $\mathcal{V}_{\text{logdet}}(\mathbf{W})$ can be viewed as a nonconvex surrogate for the rank function of $\mathbf{W}$, i.e., the L_0-norm of $\boldsymbol{\sigma}(\mathbf{W})$, up to constant factors [33, 32]. Expressed as function of singular value, the function $\mathcal{V}_{\text{logdet}}(\mathbf{W})$ with a sufficiently small δ is much closer to the L_0-norm of $\boldsymbol{\sigma}(\mathbf{W})$ than the L_1-norm of $\boldsymbol{\sigma}(\mathbf{W})$; see Fig.2.1 . Hence, if one wants to promote rank-deficient solutions, δ should not be chosen too large, say $\delta \leq 1$. Moreover, δ should not be chosen too small, otherwise:

- it could make $\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}$ badly conditioned which makes the optimization problem harder to solve, this will be explained more in §2.2 .

- it could give too much importance to zero singular values which might not be desirable. In fact, for $\sigma_i = 0$, it gives $\log\left(\sigma_i^2 + \delta\right) = \log\delta < 0$ for $\delta < 1$, which contributes to negative values in $\mathcal{V}_{\text{logdet}}$.

In practice, we recommend to use a value of $\delta \in [10^{-3}, 0.1]$ for the model with normalized $\mathbf{W}$, i.e., model (2.1b). For model (2.1a) with $\mathbf{H}$ normalized, δ can selected in the same range $[10^{-3}, 0.1]$, scaled with the spectrum of $\mathbf{M}$, say $\sigma_1^2(\mathbf{M})$. As a remark, in some works in the literature, δ was chosen too small such as 10^{-8} (without considering the scaling issue) suggested by [41] which, as explained above, is not a desirable choice, at least in the rank-deficient cases. Even in the full-rank case, we argue that choosing δ too small is also not desirable since it promotes rank-deficient solutions.

As the rule of $\delta \in [10^{-3}, 0.1]$ is practically useful enough, we do not go further on the issue on tuning δ. Note that some works in the matrix completion community consider the use of the same logdet term as a low-rank promoting regularizer, and there are some automatic tuning strategies for δ, see [87] for example. However, automatically tuning this parameter in the context of minvol NMF is a topic for further research.

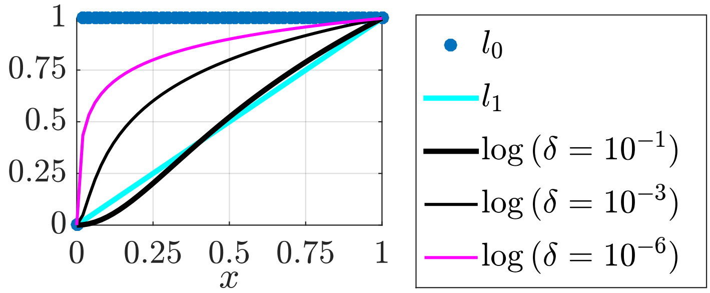


Fig. 2.1. Function $\dfrac{\log(x^2+\delta)-\log\delta}{\log(1+\delta)-\log\delta}$ for different values of δ, L_1 norm ($=|x|$) and L_0 norm (=0 for $x=0$, $=1$ otherwise). The function $\dfrac{\log(x^2+\delta)-\log\delta}{\log(1+\delta)-\log\delta}$ is the scaled and shifted version of the function $\log(x^2+\delta)$, which is the singular value function of the logdet regularizer, see Table 2.1.

The choice of λ For $\mathcal{V}_{\text{logdet}}$, the choice of δ will influence the choice of λ. In fact, the smaller δ, the larger in magnitude of logdet(δ), so to balance the two terms in the objective of minvol NMF problem (2.1a) for $\mathcal{V} = \mathcal{V}_{\text{logdet}}$, the parameter λ should be small. In applications, for rank-deficient cases, $\mathbf{W}^{(0)}, \mathbf{H}^{(0)}$ can be initialized using `SNPA` which can deal with rank-deficient SNMF problem. The value λ can then be set to

$$\lambda = \tilde{\lambda}\frac{\|\mathbf{M}-\mathbf{W}^{(0)}\mathbf{H}^{(0)}\|_F^2}{|\mathcal{V}_{\text{logdet}}(\mathbf{W}^{(0)})|},$$

where $\tilde{\lambda}$ is chosen in $[10^{-3}, 1]$ depending on the noise level (the noisier the input matrix, the larger λ should be). This expression means we pick λ that balanced the data fitting term $\|\mathbf{M}-\mathbf{W}^{(0)}\mathbf{H}^{(0)}\|_F^2$ and the regularizer $|\mathcal{V}_{\text{logdet}}(\mathbf{W}^{(0)})|$ by the $\tilde{\lambda}$ amount. In §3.3 we illustrate a simple but effective bisection search to tune λ in applying minvol NMF on imaging applications.

2.1.4 Preliminary material for the identifiability theorem

In the following two subsections, we present a new identifiability result on a minvol NMF model. To be specific, we consider the "W model" (2.1b) with $\mathcal{V}_{\text{det}}$ with the following twists:

- We removed the nonnegativity on $\mathbf{W}$, as the identifiability result does not require $\mathbf{W}$ to be nonnegative.
- Instead of regularization form as shown in (2.1b), the problem is now expressed in constrained form:

$$\underset{\mathbf{W},\mathbf{H}}{\operatorname{argmin}}\ \ \mathcal{V}_{\text{det}}(\mathbf{W}) = \det(\mathbf{W}^\top\mathbf{W}) \ \text{ subject to } \mathbf{M}=\mathbf{W}\mathbf{H},\ \mathbf{H}\geq 0,\ \mathbf{W}^\top\mathbf{1}_m = \mathbf{1}_r. \tag{2.8}$$

We now explain more on the differences between Model (2.8) and the original minvol NMF Models (2.1a) and (2.1b).

- In Models (2.1a) and (2.1b), we consider various $\mathcal{V}$; while in Model (2.8), we only focus on $\mathcal{V}(\mathbf{W}) = \det(\mathbf{W}^\top\mathbf{W})$.

- Model (2.8) has a hard constraint $\mathbf{M} = \mathbf{WH}$. Such equality constraint implies that here we assume the data $\mathbf{M}$ is noiseless and it has an exact NMF factorization. In Models (2.1a) and (2.1b), we relaxed such hard constraint to a distance function in the form of $\|\mathbf{M} - \mathbf{WH}\|_F^2$, which allows the presence of noise.
- Models (2.1a) and (2.1b) require $\mathbf{W} \geq 0$; Model (2.8) does not.
- From the optimization point of view, the hard constraint $\mathbf{M} = \mathbf{WH}$ in Model (2.8) makes the model difficult to solve efficiently.

In conclusion, Models (2.1a) and (2.1b) are more for practical purpose, and Model (2.8) is more theoretical and it gives insights on explaining why the Models (2.1a) and (2.1b) are useful in practice. When the data in Model (2.1a) satisfies the assumptions of Model (2.8), we can then use the identifiability result on Model (2.8) to explain the effectiveness of Model (2.1a).

Now we are at the position to study the identifiability of Model (2.8). That is, we ask the question: "What properties of $(\mathbf{W}, \mathbf{H})$ ensure that $(\mathbf{W}, \mathbf{H})$ is the unique minimizer of Problem (2.8)? Theorem 2.1.1 is the answer of this question, but before that, we present the background material for the theorem.

Technical background material for the identifiability theorem To study the identifiability of this model, we first give some useful definitions and lemmas for the purpose of the proof. As minvol NMF enjoys the nice geometrical interpretation of finding the smallest cone to capture the data (see §1.4.5 and §1.5), many of the definitions and lemmas below are related to the geometry of convex cones. For details of these definitions, see [9].

We begin with the definition of the cone of a matrix.

Definition 2.1.1. ***(Cone and dual cone of*** $\mathbf{W}$***)*** *The cone generated by the columns of a matrix* $\mathbf{W} \in \mathbb{R}^{m \times r}$ *is defined as*

$$\textit{cone}(\mathbf{W}) = \{\ \mathbf{x} \in \mathbb{R}^m \mid \mathbf{x} = \mathbf{Wh},\ \mathbf{h} \geq 0\ \}.$$

The dual of cone$(\mathbf{W})$*, denoted as cone*$^*(\mathbf{W})$*, is defined as*

$$\textit{cone}^*(\mathbf{W}) = \{\ \mathbf{y} \in \mathbb{R}^m \mid \langle \mathbf{x}, \mathbf{y} \rangle \geq 0, \mathbf{x} \in \textit{cone}(\mathbf{W})\ \}.$$

To characterize the the dual of cone($\mathbf{W}$), we need the following lemma.

Lemma 2.1.2. ***(Nonnegative inner product of nonnegative vectors)*** *Given a vector* $\mathbf{x}$*, then* $\mathbf{x}$ *is nonnegative if and only if* $\langle \mathbf{x}, \mathbf{h} \rangle \geq 0$ *for any* $\mathbf{h} \geq$ *0.*

Using Lemma (2.1.2) we can characterize the dual of cone($\mathbf{W}$).

Lemma 2.1.3. ***(Dual of cone of*** $\mathbf{W}$***)*** $\textit{cone}^*(\mathbf{W}) = \{\mathbf{y} \mid \mathbf{W}^\top \mathbf{y} \geq 0\}$.

Proof. We have

$$\begin{aligned} \text{cone}^*(\mathbf{W}) &= \{\ \mathbf{y} \mid \langle \mathbf{x}, \mathbf{y} \rangle \geq 0, \mathbf{x} \in \text{cone}(\mathbf{W})\ \} && \text{by Definition 2.1.1} \\ &= \{\ \mathbf{y} \mid \langle \mathbf{Wh}, \mathbf{y} \rangle \geq 0, \mathbf{h} \geq 0\ \} && \text{by Definition 2.1.1} \\ &= \{\ \mathbf{y} \mid \langle \mathbf{h}, \mathbf{W}^\top \mathbf{y} \rangle \geq 0, \mathbf{h} \geq 0\ \} \\ &= \{\ \mathbf{y} \mid \mathbf{W}^\top \mathbf{y} \geq 0\ \}. && \text{by Lemma (2.1.2)} \end{aligned}$$

□

Next, we give the definition of a standard second-order cone.

Definition 2.1.2. ***(Second-order cone)*** *The second-order cone in $\mathbb{R}^r$, denoted as $\mathcal{C}_2^r$, is defined as $\mathcal{C}_2^r = \{ \ \mathbf{x} \in \mathbb{R}_+^r \mid \langle \mathbf{1}_r, \mathbf{x} \rangle \geq \sqrt{r-1} \|\mathbf{x}\|_2 \ \}$.*

The following lemma characterizes the dual of the second-order cone.

Lemma 2.1.4. ***(Dual of second-order cone) [45]*** *The dual of $\mathcal{C}_2^r$, denoted as $\mathcal{C}_2^{r*}$ is the set $\{ \ \mathbf{x} \in \mathbb{R}^r \mid \langle \mathbf{1}_r, \mathbf{x} \rangle \geq \|\mathbf{x}\|_2 \ \}$.*

We need three more lemmas for the proof of the identifiability. Based on the duality of cone, we first have the following characterization of the nonnegativity of matrix product.

Lemma 2.1.5. ***(Conic characterization of nonnegativity of matrix product)*** *Given two matrices $\mathbf{A} \in \mathbb{R}^{m \times r}$ and $\mathbf{B} \in \mathbb{R}^{r \times n}$. Then $\mathbf{AB} \geq 0$ if cone$(\mathbf{A}^\top) \subseteq$ cone$^*(\mathbf{B})$.*

Proof. Starting from $\mathbf{AB} \geq 0 \iff \mathbf{B}^\top \mathbf{A}^\top \geq 0$, then

$$\begin{array}{rcll} \mathbf{B}^\top \mathbf{A}^\top \geq 0 & \iff & \text{the vector } \mathbf{B}^\top \mathbf{A}(i,:)^\top \geq 0, \ \forall i \in [r] & \\ & \iff & \mathbf{A}(i,:)^\top \in \{ \ \mathbf{y} \mid \mathbf{B}^\top \mathbf{y} \geq 0 \ \}, \ \forall i \in [r] & \\ & \iff & \mathbf{A}(i,:)^\top \in \text{cone}^*(\mathbf{B}), \ \forall i \in [r] & \text{by definition of dual cone of } \mathbf{B} \\ & \iff & \text{cone}(\mathbf{A}^\top) \subseteq \text{cone}^*(\mathbf{B}). & \text{by definition of cone of } \mathbf{A} \end{array}$$

□

The following lemma is about the cone of a orthogonal matrix.

Lemma 2.1.6. ***(Cone of orthogonal matrix is self-dual)*** *If matrix $\mathbf{Q}$ is orthogonal, then cone$^*(\mathbf{Q}^\top) =$ cone$(\mathbf{Q}^\top)$.*

Proof. Note that $\mathbf{Q}^\top = \mathbf{Q}^{-1}$. We have

$$\begin{array}{rcll} \text{cone}^*(\mathbf{Q}^\top) & = & \{ \ \mathbf{y} \mid (\mathbf{Q}^\top)^\top \mathbf{y} \geq 0 \ \} & \text{by Lemma (2.1.3)} \\ & = & \{ \ \mathbf{y} \mid \mathbf{Q}\mathbf{y} \geq 0 \ \} & \\ & = & \{ \ \mathbf{Q}^{-1}\mathbf{x} \mid \mathbf{Q}(\mathbf{Q}^{-1}\mathbf{x}) \geq 0 \ \} & \text{let } \mathbf{y} = \mathbf{Q}^{-1}\mathbf{x} \\ & = & \{ \ \mathbf{Q}^\top \mathbf{x} \mid \mathbf{x} \geq 0 \ \} & \mathbf{Q}\mathbf{Q}^{-1} = \mathbf{I} \text{ and } \mathbf{Q}^\top = \mathbf{Q}^{-1} \\ & = & \text{cone}(\mathbf{Q}^\top). & \text{by Definition (2.1.1)} \end{array}$$

□

The following is a standard duality result on cone inclusion.

Lemma 2.1.7. ***(Duality of cone inclusion) [9, Proposition 6.24]*** *Given two convex cones C and D. If $D \subseteq C$ then $C^* \subseteq D^*$.*

Lastly, we define the *Sufficiently Scattered Condition* (SSC) introduced by [77, 42]:

Definition 2.1.3. ***(Sufficiently scattered condition)****. A matrix $\mathbf{H} \in \mathbb{R}_+^{r \times n}$ fulfills SSC if*

- *(SSC1) cone$(\mathbf{H})$ is widespread such that it contains $\mathcal{C}_2^r$. Mathematically, $\mathcal{C}_2^r \subseteq$ cone$(\mathbf{H})$, and*
- *(SSC2) There does not exist any orthogonal matrix $\mathbf{Q}$ such that cone$(\mathbf{H}) \subseteq$ cone$(\mathbf{Q})$, except for $\mathbf{Q}$ being a permutation matrix.*

Understanding the SSC condition Geometrically, the SSC condition means that

- SSC1 : the columns of $\mathbf{H}$ are "wide spread enough" in the nonnegative orthant.
- SSC2 : cone($\mathbf{H}$) is not "too small" to be contained inside some cone($\mathbf{Q}$) where $\mathbf{Q}$ is any orthogonal matrix that is not a permutation. That is, cone($\mathbf{H}$) is "big enough" that it cannot be contained inside cone($\mathbf{Q}$).

This translate to the data points in $\mathbf{M}$, which is generated as $\mathbf{WH}$, are sufficiently wide spread in cone($\mathbf{W}$), see Fig.2.2 for an illustration.

As illustrated in Fig.2.2 , the key of SSC condition is that the data cone is widespread so that it contains the second-order cone $\mathcal{C}_2$. This translates to a specific sparsity pattern on $\mathbf{H}$. Considering the outermost data points of $\mathbf{M}$ in the data cone in Fig.2.2 , the matrix $\mathbf{H}$ that generates this data has the following structure:

$$\mathbf{H}_{\text{SSC}} = \begin{bmatrix} c+\epsilon & 1-c-\epsilon & c+\epsilon & 1-c-\epsilon & 0 & 0 \\ 1-c-\epsilon & c+\epsilon & 0 & 0 & c+\epsilon & 1-c-\epsilon \\ 0 & 0 & 1-c-\epsilon & c+\epsilon & 1-c-\epsilon & c+\epsilon \end{bmatrix}, \tag{2.9}$$

where $c = \dfrac{1}{\sqrt{2}}$ and $\epsilon > 0$ such that $\mathbf{H}_{\text{SSC}} \geq 0$. That is, in the case $r = 3$, if $\mathbf{H}$ has these six columns in $\mathbf{H}_{\text{SSC}}$, then the data $\mathbf{M}$ generated by such an $\mathbf{H}$ is widespread so that fitting a minimum-volume polytope on the data will recover the ground truth vertices that generate the data. In other words, the SSC condition implies some specific sparsity structure in $\mathbf{H}$ (see Equation (2.9) as an example), and once $\mathbf{H}$ satisfies such sparsity structure, together with some other conditions, the minvol NMF problem is identifiable.

Checking the SSC condition A natural question concerning the SSC condition is that: "how to know if the data fulfill the SSC condition?". Unfortunately, it is NP-hard to check whether the data satisfy the SSC condition [56] .

2.1.5 Identifiability of minvol NMF using determinant volume

With the geometric intuition of SSC, we now present the identifiability of solving the minvol NMF problem (2.8). This is one of the theoretical contribution of the thesis. The following material appeared in [72].

Theorem 2.1.1. *Let* $\mathbf{M} = \mathbf{WH}$ *where* $\mathbf{H} \in \mathbb{R}_+^{r\times n}$ *satisfies the SSC,* $\mathbf{W} \in \mathbb{R}^{m\times r}$ *satisfies* $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$ *and rank*$(\mathbf{M}) = r$*. Then the (exact) solution of the minvol NMF problem* (2.8) *is unique.*

Proof. By contradiction. Let $(\bar{\mathbf{W}}, \bar{\mathbf{H}})$ be a feasible (exact) solution of the minvol NMF problem (2.8) with $\mathcal{V} = \mathcal{V}_{\text{det}}$, suppose for the purpose of contradiction, there exists another pair of feasible solution $(\mathbf{W}^\#, \mathbf{H}^\#)$.

The first step is to link $\bar{\mathbf{W}}$ and $\mathbf{W}^\#$. By $\mathbf{M} = \bar{\mathbf{W}}\bar{\mathbf{H}} = \mathbf{W}^\#\mathbf{H}^\#$ and rank $(\mathbf{M}) = r$, so

$$r = \text{rank}\,(\mathbf{M}) \leq \min\{\text{rank}\,(\bar{\mathbf{W}}), \text{rank}\,(\bar{\mathbf{H}})\},$$

so rank $(\bar{\mathbf{W}})$ is at least r. As $\bar{\mathbf{W}}$ has r columns, so $\bar{\mathbf{W}}$ is full rank. The same conclusion holds for $\mathbf{W}^\#$. Hence there exists an r-by-r invertible matrix $\mathbf{Q}$ such that

$$\bar{\mathbf{W}} = \mathbf{W}^\#\mathbf{Q}^{-1}, \quad \bar{\mathbf{H}} = \mathbf{Q}\mathbf{H}^\#. \tag{2.10}$$

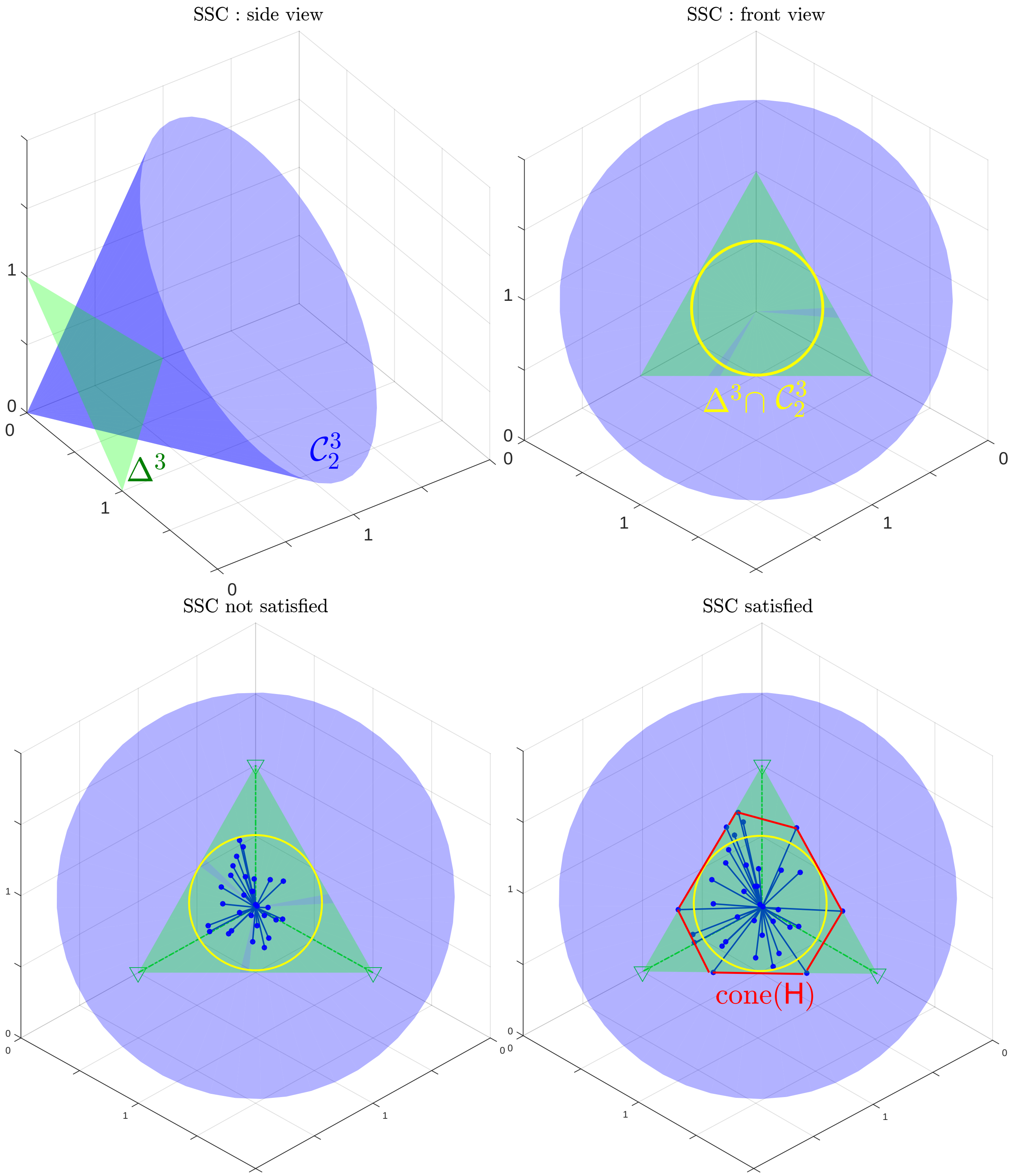


Fig. 2.2. A 3-dimensional example to illustrate SSC. Here $r = 3$. The unit simplex Δ^3 is the green triangle, and the second-order cone $\mathcal{C}_2^3$ is indicated in blue. **Top row**: illustration of Δ^3 and $\mathcal{C}_2^3$ and the boundary of the disk $\mathcal{C}_2^r \cap \Delta^r$ (which is the intersection of the second-order cone $\mathcal{C}_2^r$ and the simplex Δ^r, where the boundary of this disc is illustrated as the yellow circle in the figure).

Bottom row: example of data that do not and do satisfy the SSC. For data to satisfy SSC, the data has to be widespread so that the convex hull of these points (the red polygon) contains the disk $\mathcal{C}_2^r \cap \Delta^r$.

Then

$$\mathcal{V}(\bar{\mathbf{W}}) = \det(\bar{\mathbf{W}}^\top \bar{\mathbf{W}}) \overset{(2.10)}{=} |\det(\mathbf{Q})|^{-2} \det(\mathbf{W}^{\#^\top} \mathbf{W}^{\#}) = |\det(\mathbf{Q})|^{-2} \mathcal{V}(\mathbf{W}^{\#}).$$

In the remaining of the proof, we show $|\det(\mathbf{Q})| < 1$, which leads to $\mathcal{V}(\bar{\mathbf{W}}) \neq \mathcal{V}(\mathbf{W}^{\#})$, hence a contradiction on the optimality of the solutions $\bar{\mathbf{W}}$ and $\mathbf{W}^{\#}$, and thereby the solution of the minvol NMF problem (2.8) is unique.

From the assumption,

$$\bar{\mathbf{W}}^\top \mathbf{1}_m = \mathbf{W}^{\#^\top} \mathbf{1}_m = \mathbf{1}_r. \tag{2.11}$$

Then,

$$\mathbf{1}_r \overset{(2.11)}{=} \bar{\mathbf{W}}^\top \mathbf{1}_m \overset{(2.10)}{=} \mathbf{Q}^{-\top} \mathbf{W}^{\#^\top} \mathbf{1}_m \overset{(2.11)}{=} \mathbf{Q}^{-\top} \mathbf{1}_r,$$

which gives

$$\mathbf{Q}^\top \mathbf{1}_r = \mathbf{1}_r. \tag{2.12}$$

The equality (2.12) gives us some information on $\mathbf{Q}$, based on the assumptions on $\bar{\mathbf{W}}$ and $\mathbf{W}^{\#}$. Now we derive another piece of information on $\mathbf{Q}$ using the assumptions on $\bar{\mathbf{H}}$ and $\mathbf{H}^{\#}$. First $\bar{\mathbf{H}} \overset{(2.10)}{=} \mathbf{Q}\mathbf{H}^{\#} \geq 0$, by Lemma 2.1.5,

$$\text{cone}(\mathbf{Q}^\top) \subseteq \text{cone}^*(\mathbf{H}^{\#}). \tag{2.13}$$

Now, as $\mathbf{H}^{\#}$ satisfies SSC, so $\mathcal{C}_2^r \subseteq \text{cone}(\mathbf{H}^{\#})$, then by duality of inclusion (Lemma (2.1.7))

$$\text{cone}^*(\mathbf{H}^{\#}) \subseteq \mathcal{C}_2^{r*}. \tag{2.14}$$

Put (2.14) into (2.13) gives $\text{cone}(\mathbf{Q}^\top) \subseteq \mathcal{C}_2^{r*}$. Using Lemma 2.1.4 to characterize $\mathcal{C}_2^{r*}$ gives

$$\langle \mathbf{1}_r, \mathbf{Q}(i,:) \rangle \geq \|\mathbf{Q}(i,:)\|_2. \tag{2.15}$$

Now we are at the position to show $|\det(\mathbf{Q})| < 1$.

$$\begin{aligned}
|\det(\mathbf{Q})| &= |\det(\mathbf{Q}^\top)| \\
&\leq \textstyle\prod_{i=1}^r \|\mathbf{Q}(i,:)\|_2 && \text{Hardmard inequality [50]} \\
&\leq \textstyle\prod_{i=1}^r \langle \mathbf{1}_r, \mathbf{Q}(i,:) \rangle && \text{by Inequality (2.15)} \\
&\leq \left(\frac{\sum_{i=1}^r \langle \mathbf{1}_r, \mathbf{Q}(i,:) \rangle}{r} \right)^r && \text{Inequality of arithmetic and geometric means} \\
&= \left(\frac{\langle \mathbf{1}_r, \mathbf{Q}\mathbf{1}_r \rangle}{r} \right)^r \\
&= 1. && \text{by Equation (2.12)}
\end{aligned}$$

Hence $|\det(\mathbf{Q})| \leq 1$, which gives two cases, $|\det(\mathbf{Q})| < 1$ or $|\det(\mathbf{Q})| = 1$. If we are in the first case, the proof is completed.

Now we consider the second case and show $\mathbf{Q} = \mathbf{\Pi}$ as follows. First the equality in $|\det(\mathbf{Q})| = 1$ gives $\|\mathbf{Q}(i,:)\|_2 = \langle \mathbf{1}_r, \mathbf{Q}(i,:) \rangle$ for all i, and also the Hadamard's inequality is now achieved as $|\det(\mathbf{Q})| = \prod_{i=1}^r \|\mathbf{Q}(i,:)\|_2$. By the fact that Hadamard's inequality is achieved if the vectors are orthogonal, so $\mathbf{Q}$ is orthogonal.

From (2.13), applying duality of inclusion (Lemma (2.1.7)) gives $\text{cone}(\mathbf{H}^{\#}) \subseteq \text{cone}^*(\mathbf{Q}^\top)$. Then by Lemma (2.1.6) we get $\text{cone}(\mathbf{H}^{\#}) \subseteq \text{cone}(\mathbf{Q}^\top)$. As $\mathbf{H}^{\#}$ satisfies SSC (Definition 2.1.3), so $\mathbf{Q} = \mathbf{\Pi}$. □

Let's us discuss more on Theorem 2.1.1.

- The theorem assumes noiseless condition and exact NMF. When the data is noisy, we go back to the minvol NMF formulation (2.1a), where the regularization parameter λ is chosen depending on the noise level (see the discussion right before §2.1.5).

- The theorem is slightly more general than standard NMF setting. The constraint on $\mathbf{W}$ in the theorem contains only normalization $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$, there is no nonnegativity constraint on $\mathbf{W}$. For NMF without the nonnegativity constraint on $\mathbf{W}$, the model is called Semi-NMF, which is useful in a variety of other applications, but it is not the focus of this thesis. Note that if $\mathbf{W}$ is nonnegative, we assume without loss of generality that $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$, as $\mathbf{W}$ is full rank and we can absorb the scaling constants in $\mathbf{H}$.

- The theorem is new, although the constraint $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$ in minvol NMF was historically first considered in [114], but they did not provide any identifiability result. Furthermore, Theorem 2.1.1 is different from existing theorems (see below) with similar proof structure, where the difference is that Theorem 2.1.1 considers normalization on $\mathbf{W}$ while existing theorems consider normalization on $\mathbf{H}$.

- Comparing the normalization of the columns of $\mathbf{W}$ in Theorem 2.1.1 with other modes of normalization in minvol NMF: $\mathbf{H}^\top \mathbf{1}_r = \mathbf{1}_n$ (where the identifiability result is provided by [77, 42]), $\mathbf{H}\mathbf{1}_n = \mathbf{1}_r$ (where identifiability result is provided by [39]), all the minvol NMF models with these normalization lead to the same identifiability result in the absence of noise, but in practice the normalization of the columns of $\mathbf{W}$ is better:
  - It balances better the relative importance of each columns of $\mathbf{W}$ in the volume regularization term since now all the norm of these columns are bounded.
  - It provides $\mathbf{W}$ that are better conditioned and hence leads to more stable numerical algorithms. This is observed on many numerical examples. Empirically we observed that the normalization $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$ is much less sensitive to noise and returns much better solutions. The reasons are:
    (i) Using the normalization $\mathbf{H}\mathbf{1}_n = \mathbf{1}_r$ amounts to multiply $\mathbf{W}$ by a diagonal matrix whose entries are the L_1 norms of the rows of $\mathbf{H}$. So, the columns of $\mathbf{W}$ that correspond to dominating (resp. dominated) component (the rank-1 factor $\mathbf{w}_i\mathbf{h}^i$), will have much higher (resp. lower) norm. Therefore, the term $\mathbf{W}^\top\mathbf{W}$ (and thereby both $\mathcal{V}_{\text{det}}(\mathbf{W})$ and $\mathcal{V}_{\text{logdet}}(\mathbf{W})$) is much more influenced by the dominating component and will have difficulties to penalize the dominated components in terms of both the algorithm and the model. In other words, the use of $\mathcal{V}_{\text{det}}(\mathbf{W})$ and the normalization $\mathbf{H}\mathbf{1}_n = \mathbf{1}_r$ implicitly requires that the rank-1 factors $\mathbf{w}_i\mathbf{h}^i$ for $i \in [r]$ are all well balanced, i.e., have similar norms. This is not the case for many real data.
    (ii) In §2.1.3 , on the choice of δ, we discussed that the numerical stability of updating $\mathbf{W}$ is related to the condition number of $\mathbf{W}^\top\mathbf{W}$. Based on the normalization on $\mathbf{W}$, we can bound the condition number as follows

$$\begin{aligned}\kappa(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_r) = \frac{\sigma_{\min}(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_r)}{\sigma_{\min}(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_r)} &= \frac{\big(\sigma_{\max}(\mathbf{W})\big)^2 + \delta}{\big(\sigma_{\min}(\mathbf{W})\big)^2 + \delta} \\ &\leq \frac{\big(\sqrt{r}\max_j \|\mathbf{W}(:,j)\|_2\big)^2 + \delta}{\delta} \quad \leq 1 + \frac{r}{\delta},\end{aligned}$$

where the first inequality comes from the following standard norm inequality:

$$\sigma_{\max}(\mathbf{W}) = \|\mathbf{W}\|_2 \leq \|\mathbf{W}\|_F = \Big(\sum_{j=1}^{r} \|\mathbf{W}(:,j)\|_2^2\Big)^{\frac{1}{2}} \leq \Big(r \max_j \|\mathbf{W}(:,j)\|_2^2\Big)^{\frac{1}{2}}.$$

On the other hand, the normalization $\mathbf{H}\mathbf{1}_n = \mathbf{1}_r$ may lead to arbitrarily large values for the condition number of $\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_r$, which it has been observed numerically on several examples.

2.2 Solving minvol NMF

Here we focus how to numerically solve problem (2.1a) , which is the model used for hyperspectral imaging application in the next chapter. We will discuss how to numerically solve problem (2.1b), which is the model used for analytical chemistry application in §5.1.3 .

We restate problem (2.1a) here for convenience: given $\mathbf{M} \in \mathbb{R}_+^{m\times n}$ and a factorization rank $r \in \mathbb{N}$, find $\mathbf{W} \in \mathbb{R}_+^{m\times r}$ and $\mathbf{H} \in \mathbb{R}_+^{r\times n}$ by solving

$$\text{minimize } \ \frac{1}{2}\|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 + \lambda\mathcal{V}(\mathbf{W}) \ \text{ subject to } \ \mathbf{W} \geq 0, \mathbf{H} \geq 0, \mathbf{H}^\top\mathbf{1}_r \leq \mathbf{1}_n.$$

The basic approach to solve this problem is *Alternating Minimization*, similar to Algorithm 1 , we solve the subproblem on $\mathbf{H}$ while fixing $\mathbf{W}$, then solve the subproblem on $\mathbf{W}$ while fixing $\mathbf{H}$. When fixing a variable, the latest version of that variable is used. In the following we discuss how to solve the two subproblems.

2.2.1 Solving subproblem on H

Splitting $\mathbf{H}$ in problem (2.1a) into columns yields n independent subproblems: for all $j \in [n]$, solve

$$\operatorname*{argmin}_{\mathbf{h}_j} \ \frac{1}{2}\|\mathbf{W}\mathbf{h}_j - \mathbf{m}_j\|_2^2 \ \text{ s.t. } \mathbf{h}_j \in \Delta^r := \{\mathbf{h}_j \in \mathbb{R}_+^r \mid \mathbf{h}_j^\top\mathbf{1}_r \leq 1\}, \tag{2.16}$$

where $\mathbf{h}_j$ is the jth column of $\mathbf{H}$ and $\mathbf{m}_j$ denotes the jth column of $\mathbf{M}$. Let rank $(\mathbf{W}) = r$, which is a standard assumption, this least squares problem over the unit simplex is a convex problem with strongly convex cost function. Solving it using Accelerated Projected Gradient (AccProjG) with Nesterov's acceleration [89] takes $\mathcal{O}\Big(\sqrt{\kappa}\log\frac{1}{\epsilon}\Big)$ iterations to reach an ϵ-accurate solution, where κ is the condition number of $\mathbf{W}^\top\mathbf{W}$, which is the Hessian of the cost function in problem (2.16). The convergence rate of AccProjG is optimal as no other 1st-order method can have a faster convergence rate [89].

The key operation in AccProjG is the projection onto the set Δ^r, which is a convex set and therefore the projection solution exists and is unique. Below we give the solution to this projection; we move the technical details here to §5.1.7 , where we consider the projection onto irregular simplex. The projection onto Δ^r is

$$P_{\Delta^r}(\mathbf{h}_j) = [\mathbf{h}_j - \tau\mathbf{1}_r]_+, \tag{2.17}$$

where τ is a Lagrangian multiplier associated to the inequality constraint $\mathbf{h}_j^\top\mathbf{1}_r \leq 1$. We can see that, $P_{\Delta^r}(\mathbf{h}_j)$ is basically a thresholding operation, where the main difficulty here is to find the correct value of τ. In the paper [6, 4], we use the implementation from [43], which is a sorting-based method to compute τ, and the overall complexity is $\mathcal{O}(r\log r)$ operations. As mentioned by [24], this is in fact suboptimal, and the optimal method introduced in [24] has the complexity $\mathcal{O}(r)$. In practice,

r is usually small (say $r \leq 20$), and therefore the suboptimal method is numerically as good as the optimal one.

Remark 1. *Lastly, we give two remarks on solving problem* (2.16)*.*

- *Problem* (2.16) *involves the variables* $(\mathbf{W}, \mathbf{h}_j, \mathbf{m}_j)$*, where* $\mathbf{h}_j$ *is independent of other columns in* $\mathbf{H}$ *(and similarly for* $\mathbf{m}_j$*), this problem is separable and can be solved in parallel.*
- *The convergence rate of AccProjG is* $\mathcal{O}\Big(\sqrt{\kappa}\log\frac{1}{\epsilon}\Big)$*, which is strongly dependent on the condition number of* $\mathbf{W}^\top\mathbf{W}$*.*
  - *As mentioned in §2.1.3, a poor choice of parameter (such as* δ *in* $\mathcal{V}_{logdet}$*) that leads to a poor condition of* $\mathbf{W}^\top\mathbf{W}$*, can seriously slow down the optimization of* $\mathbf{H}$*. In fact, this is one of the reason why the method* `RVolMin` *[41], a state-of-the-art minvol model that closely related to minvol NMF, was reported in [6] that it performs weaker than minvol NMF. We will report the experiment section of [6] in §3.3.*
  - *In rank-deficient situation* $\kappa(\mathbf{W}^\top\mathbf{W}) = \infty$*, using AccProjG to solve problem* (2.16) *will have much slower convergence, it will be sublinear as* $\mathcal{O}(\frac{1}{k^2})$*, where* k *is the iteration number.*
- *We will discuss a generalizations of the Problem* (2.16) *later in §3.3.3, where we consider the problem with additional sparsity regularizer. In §5.1.7, we consider the projection onto irregular simplex.*
- *It is possible to solve Problem* (2.16) *using other methods, for example the interior point method, but in this thesis we will stick with the first-order method AccProjG.*

2.2.2 Solving subproblem on W

The subproblem on $\mathbf{W}$ has the form

$$\underset{\mathbf{W}}{\operatorname{argmin}} \ \frac{1}{2}\|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 + \lambda\mathcal{V}(\mathbf{W}) \ \text{ s.t. } \mathbf{W} \geq 0. \tag{2.18}$$

Recall the $\mathcal{V}(\mathbf{W})$ in Table 2.1 can be expressed as $\mathcal{V}(\mathbf{W}) = g \circ \boldsymbol{\sigma}(\mathbf{W})$, which immediately delivers us a systematic approach to solve subproblem (2.18) using subgradient methods. We now give a short review on this approach below.

Subgradient approach First, we recall subgradient and subdifferential, see [10] for more details.

Definition 2.2.1. ***(Subgradient and Subdifferential)*** *Let* $f(\mathbf{x}) : \mathbb{R}^n \to]-\infty, +\infty]$ *be a proper function and let* $\mathbf{x} \in$*domf. A vector* $\mathbf{g} \in \mathbb{R}^n$ *is called a subgradient of* f *at* $\mathbf{x}$ *if*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \mathbf{g}, \mathbf{y} - \mathbf{x}\rangle \quad \textit{for all } \mathbf{y} \in \mathbb{R}^n.$$

The subdifferential is the set of all subgradients of f *at* $\mathbf{x}$*.*

To use subgradient to solve the subproblem, the update is a projected subgradient step:

$$\mathbf{W}_{k+1} = \Big[\mathbf{W}_k - \alpha\mathbf{G}_{\mathcal{V}}(\mathbf{W})\Big]_+ \tag{2.19}$$

where $\alpha > 0$ is a stepsize, $\mathbf{G}_{\mathcal{V}}(\mathbf{W})$ is a subgradient of $\mathcal{V}(\mathbf{W})$ at the point $\mathbf{W}$, and $[\ \cdot\ ]_+ = \max\{0, \cdot\}$ is the projection. For the $\mathcal{V}$ function listed in Table 2.1 , the subgradient can be computed using the following formula [74, Theorem 7.1]:

$$\partial_{\mathbf{W}}\mathcal{V}(\mathbf{W}) = \mathbf{U}\mathrm{Diag}\Big\{\partial_{\boldsymbol{\sigma}}(g \circ \boldsymbol{\sigma}(\mathbf{W}))\Big\}\mathbf{V}^\top,$$

where $\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ is the SVD of $\mathbf{W}$ and $\boldsymbol{\Sigma} = \mathrm{Diag}\{\boldsymbol{\sigma}\}$.

For example, for the Frobenius norm-squared $\mathcal{V}_\mathrm{F} = \|\mathbf{W}\|_F^2$,

$$\partial_{\mathbf{W}}\mathcal{V}_\mathrm{F}(\mathbf{W}) = \mathbf{U}\mathrm{Diag}\Big\{\partial_{\boldsymbol{\sigma}}\Big(\sum_i \sigma_i^2\Big)\Big\}\mathbf{V}^\top = 2\mathbf{U}\mathrm{Diag}\{\boldsymbol{\sigma}\}\mathbf{V}^\top = 2\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top = 2\mathbf{W},$$

which agrees with the fact that $\partial_{\mathbf{W}}\|\mathbf{W}\|_F^2 = 2\mathbf{W}$. For $\mathcal{V}_\mathrm{logdet}$,

$$\partial_{\mathbf{W}}\mathcal{V}_\mathrm{logdet}(\mathbf{W}) = \mathbf{U}\mathrm{Diag}\Big\{\partial_{\boldsymbol{\sigma}}\Big(\sum_i \log(\sigma_i^2 + \delta)\Big)\Big\}\mathbf{V}^\top, = 2\mathbf{U}\mathrm{Diag}\Big\{\frac{\sigma_i}{\sigma_i^2 + \delta}\Big\}\mathbf{V}^\top,$$

in which there is no a closed-form expression in terms of matrix $\mathbf{W}$.

We now have a systematic approach for solving all the minvol NMF problems for $\mathcal{V}$ in Table 2.1 . However, such approach has several drawbacks: 1) it is well known that subgradient methods are slow. 2) Some cost functions in these subproblems are nonconvex, in which subgradient methods may not exist. Thus, to solve the subproblems on $\mathbf{W}$, in the following we consider different methods that are specific to each $\mathcal{V}$.

On $\mathcal{V}_\mathbf{det}$: using quadratic form We follow the approach used in [114] to solve the subproblem on $\mathbf{W}$. Using the quadratic form (2.6), we split $\mathbf{W}$ in problem (2.18) into r columns, and minimize with respect to each column one-by-one. First

$$\|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 = \|\mathbf{h}^i\|_2^2\|\mathbf{w}\|_2^2 - 2\langle \mathbf{M}_i{\mathbf{h}^i}^\top, \mathbf{w}_i\rangle + c, \tag{2.20}$$

where $\mathbf{M}_i = \mathbf{M} - \sum_{j\neq i}\mathbf{w}_j\mathbf{h}^j$ and c is a constant independent of $\mathbf{w}_i$. Using equality (2.20) together with the quadratic form (2.6), we rewrite Problem (2.18) as the following constrained Quadratic Programming (QP) problem

$$\underset{\mathbf{w}_j \geq 0}{\mathrm{argmin}}\ \frac{1}{2}\mathbf{w}_i^\top\Big(\|\mathbf{h}^i\|_2^2\mathbf{I}_m + \det(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})(\mathbf{I}_m - \mathbf{W}_{-i}(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})^{-1}\mathbf{W}_{-i}^\top)\Big)\mathbf{w}_i - \langle \mathbf{M}_i{\mathbf{h}^i}^\top, \mathbf{w}_i\rangle. \tag{2.21}$$

Unlike the problem on $\mathbf{h}$, here the objective function in $\mathbf{w}_i$ depends on the other columns in $\mathbf{W}$ so problem (2.21) is not separable (i.e., they cannot be solved in parallel directly). This QP can then be solved by any QP solver, or using AccProjG. Here the major computational cost comes from computing the coefficient of the QP, especially the term:

$$\boldsymbol{\Theta}_i = \|\mathbf{h}^i\|_2^2\mathbf{I}_m + \det(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})(\mathbf{I}_m - \mathbf{W}_{-i}(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})^{-1}\mathbf{W}_{-i}^\top).$$

The diagonal matrix $\|\mathbf{h}^i\|_2^2\mathbf{I}_m$ is not expensive to compute, but what matters is the second part in $\boldsymbol{\Theta}_i$. When the dimension m is not large, $\boldsymbol{\Theta}_i$ can be computed as shown directly, which costs about $\mathcal{O}(3m^3)$ in terms of time complexity. When m is large, $\boldsymbol{\Theta}_i$ can be computed using Singular Value Decomposition (SVD) as follows [114]

1. Run SVD to obtain the left singular vectors of $\mathbf{W}_{-i}$. i.e., $\mathbf{U} \leftarrow$ SVD $(\mathbf{W}_{-i})$.

2. Take only the last $m-r$ columns in $\mathbf{U}$, i.e., $\mathbf{U}(:,r,r+1,\dots,m)$.

3. Compute the matrix $\boldsymbol{\Theta}_i$ as

$$\boldsymbol{\Theta}_i = \|\mathbf{h}^i\|_2^2\mathbf{I}_m + \det(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})\mathbf{U}(:,r,r+1,\dots,m)\mathbf{U}(:,r,r+1,\dots,m)^\top.$$

The computational cost is slightly better as it runs in $\mathcal{O}(3m^3 - m^2r)$. When m is huge, both approaches essentially share the same complexity in $\mathcal{O}(m^3)$, but the SVD approach is numerically more robust.

On $\mathcal{V}_{\mathsf{logdet}}(\mathbf{W})$: by majorization minimization To solve this subproblem, we use Majorization Minimization (MM) as in [6, 4]. The idea of MM is that, at iteration k, instead of minimizing $f(\mathbf{x})$, we build a tight upper bound called majorizer $g_k(\mathbf{x};\theta_k)$ of f, where θ_k denotes some parameters. We minimize g_k at the iteration k, and then we update the parameter θ_k. As g_k is always an upper bound of f, thus we are indirectly minimizing f in MM. Such approach is guaranteed to converge, even in block-variable setting; see [93].

First we derive the following inequality

Lemma 2.2.1. *(Majorizer of logdet of Gram matrix) Given a matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$, a constant $\delta \geq 0$, then for any matrix $\mathbf{Z} \in \mathbb{R}^{m\times n}$, the following inequality holds*

$$\text{logdet}(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_n) \leq \text{logdet}(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n) + \text{Tr}\left((\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n)^{-1}\mathbf{W}^\top\mathbf{W}\right) + \sum_{i=1}^n \frac{\delta}{\sigma_i^2(\mathbf{Z}) + \delta} - n,$$

and the equality holds if $\mathbf{Z} = \mathbf{W}$.

Proof. Let $f(\mathbf{X}) = \text{logdet}(\mathbf{X})$. It is a concave function in $\mathbf{X}$, and we can upper bound it by its first-order Taylor approximation as $f(\mathbf{X}) \leq f(\mathbf{Y}) + \langle\nabla f(\mathbf{Y}), \mathbf{X} - Y\rangle$, where $\nabla f(\mathbf{X}) = \mathbf{X}^{-\top}$, so

$$f(\mathbf{X}) \leq f(\mathbf{Y}) + \langle\mathbf{Y}^{-\top}, \mathbf{X} - \mathbf{Y}\rangle = f(\mathbf{Y}) + \text{Tr}\left(\mathbf{Y}^{-1}\mathbf{X}\right) - n.$$

Now substitute $\mathbf{X} = \mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_n$ and $\mathbf{Y} = \mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n$ gives

$$\begin{aligned} & \text{logdet}\left(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_n\right) \\ \leq\ & \text{logdet}\left(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n\right) + \text{Tr}\left(\left(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n\right)^{-1}\left(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_n\right)\right) - n \\ =\ & \text{logdet}\left(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n\right) + \text{Tr}\left(\left(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n\right)^{-1}\mathbf{W}^\top\mathbf{W}\right) + \delta\text{Tr}\left(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n\right)^{-1} - n. \end{aligned}$$

Note that $\mathbf{Z} = \mathbf{W}$ gives $\mathbf{X} = \mathbf{Y}$ and the equality is established. Our remaining task is to show that

$$\text{Tr}\left(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n\right)^{-1} = \sum_{i=1}^n \frac{1}{\sigma_i^2(\mathbf{Z}) + \delta}.$$

Let $\mathbf{U}\Sigma\mathbf{V}$ be the SVD of $\mathbf{Z}$,

$$\text{Tr}\left(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_n\right)^{-1} = \text{Tr}\left(\mathbf{V}\Sigma^2\mathbf{V}^\top + \delta\mathbf{V}\mathbf{I}_n\mathbf{V}^\top\right)^{-1} = \text{Tr}\left(\Sigma^2 + \delta\mathbf{I}_n\right)^{-1} = \sum_{i=1}^n \frac{1}{\sigma_i^2(\mathbf{Z}) + \delta}.$$

□

Using the above lemma on the regularizer in the minvol NMF,

$$\text{logdet}(\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}_r) \leq \text{Tr}\left((\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_r)^{-1}\mathbf{W}^\top\mathbf{W}\right) + c,$$

where c denotes the constant independent of $\mathbf{W}$. We can see that the trace term in the inequality is a weighted Frobenius norm with $(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_r)^{-1} \succ 0$ for any matrix $\mathbf{Z}$, $\delta > 0$, where $\mathbf{A} \succ 0$ means the matrix $\mathbf{A}$ is positive-definite. In MM, at each iteration in the algorithm, we solve the following problem

$$\begin{aligned} \mathbf{W} &= \underset{\mathbf{W}\geq 0}{\text{argmin}} \ \frac{1}{2}\langle \mathbf{W}^\top\mathbf{W}, \mathbf{H}\mathbf{H}^\top\rangle - \langle \mathbf{M}\mathbf{H}^\top, \mathbf{W}\rangle + \frac{\lambda}{2}\text{Tr}\left((\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_r)^{-1}\mathbf{W}^\top\mathbf{W}\right) \\ &= \underset{\mathbf{W}\geq 0}{\text{argmin}} \ \frac{1}{2}\text{Tr}\left(\left(\mathbf{H}\mathbf{H}^\top + \lambda(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_r)^{-1}\right)\mathbf{W}^\top\mathbf{W}\right) - \langle \mathbf{M}\mathbf{H}^\top, \mathbf{W}\rangle \end{aligned} \tag{2.22}$$

where we set $\mathbf{Z}$ as $\mathbf{W}$ in the previous iteration. Such problem is also a QP, it can be solved by AccProjG, in which the convergence now is related to the condition number of the matrix $\mathbf{H}\mathbf{H}^\top + \lambda(\mathbf{Z}^\top\mathbf{Z} + \delta\mathbf{I}_r)^{-1}$, where poorly chosen parameters λ, δ can lead to a bad condition for this matrix, and slow down the AccProjG.

Finally we mention two ways to accelerate the update of $\mathbf{W}$. First, the HER acceleration framework, which will be discussed in §7 , can be employed. Second, we can make use of the idea from [46] : notice that several terms in the subproblem are independent of $\mathbf{W}$, in particular $\mathbf{H}\mathbf{H}^\top$ and $\mathbf{M}\mathbf{H}^\top$, which form the main computational cost of the update. Hence we can (i) pre-computing these terms independent of $\mathbf{W}$ outside the update to avoid repeated computation of the same terms, and (ii) perform the update on $\mathbf{W}$ multiple times to reuse these precomputed terms (in most standard NMF algorithms, $\mathbf{W}$ is only updated once before the update of $\mathbf{H}$).

On $\mathcal{V}_*(\mathbf{W})$: a heuristic The nuclear norm regularized minvol NMF problem on $\mathbf{W}$ is

$$\underset{\mathbf{W}\geq 0}{\text{argmin}} \ \frac{1}{2}\langle \mathbf{W}^\top\mathbf{W}, \mathbf{H}\mathbf{H}^\top\rangle - \langle \mathbf{M}\mathbf{H}^\top, \mathbf{W}\rangle + \lambda\|\mathbf{W}\|_*, \tag{2.23}$$

We solve it using the following heuristics. Ignoring for now the nonnegativity constraints, we solve the resulting problem by proximal gradient that updates $\mathbf{W}$ as follows. At iteration k,

$$\mathbf{W}_{k+\frac{1}{3}} = \mathbf{W}_k - \alpha\nabla_{\mathbf{W}} f(\mathbf{W}_k), \quad \mathbf{W}_{k+\frac{2}{3}} = \text{prox}_{\lambda\|\cdot\|_*}\left\{\mathbf{W}^{k+\frac{1}{3}}\right\},$$

where α is the step size and ∇f is the gradient of $\frac{1}{2}\langle \mathbf{W}^\top\mathbf{W}, \mathbf{H}\mathbf{H}^\top\rangle - \langle \mathbf{M}\mathbf{H}^\top, \mathbf{W}\rangle$ with respect to $\mathbf{W}$. For the step size, we follows the standard in 1st-order optimization by using $\alpha = 1/L$ where $L = \|\mathbf{H}\mathbf{H}^\top\|_2$ is the Lipschitz constant of the gradient ∇f. The proximal operator of Nuclear norm has a closed-form expression given by the Singular Value Thresholding (SVT) operator [17]

$$\mathbf{W}_{k+\frac{2}{3}} = \text{SVT}_\lambda\left\{\mathbf{W}_{k+\frac{1}{3}}\right\} := \mathbf{U}[\Sigma - \lambda\mathbf{I}]_+\mathbf{V}^\top, \tag{2.24}$$

where $\mathbf{U}\Sigma\mathbf{V}^\top$ is the SVD of $\mathbf{W}_{k+\frac{1}{3}}$. Finally, to take the nonnegativity constraint into account, we simply put a nonnegative projection after SVT step (2.24) and set

$$\mathbf{W}_{k+1} = \left[\mathbf{W}_{k+\frac{2}{3}}\right]_+.$$

Note that the method is a heuristic. This heuristic works in practice (see §3.3). For designing algorithm with theoretical guarantee for minvol NMF with $\mathcal{V}_*(\mathbf{W})$ will be future work.

About computational cost As discussed in previous paragraphs:

- For minvol NMF with $\mathcal{V}_{\text{det}}$, we use BCD on the columns of $\mathbf{W}$, and the computational bottleneck is on computing the matrix $\mathbf{Q}_i$; see (2.6) . The cost per iteration can be high if m is large.

- For minvol NMF with $\mathcal{V}_{\text{logdet}}$, we use a global majorization minimization framework which is much cheaper per iteration.

- For minvol NMF with $\mathcal{V}_*$, the heuristic proposed involves the use of the SVT operator, which requires the SVD of $\mathbf{W}$. The update can be expensive if m is large.

2.2.3 About convergence

We now discuss the convergence of the sequences $\{f(\mathbf{W}_k, \mathbf{H}_k)\}_{k\in\mathbb{N}}$ and $\{\mathbf{W}_k, \mathbf{H}_k\}_{k\in\mathbb{N}}$ produced by the algorithms described in the previous subsections.

Informal discussion First we give an informal but simple and quick discussion on the convergence. In each iteration of the algorithm, we update the block variable $\mathbf{W}$ or $\mathbf{H}$ by solving a particular optimization subproblem. For these optimization subproblems, they are QPs with strongly convex objective (assuming $\mathbf{W}$ and $\mathbf{H}$ are full rank), which means that the solution to these subproblems exists and is unique. Furthermore, the cost sequence monotonically decrease as $f(\mathbf{W}_k, \mathbf{H}_k) \leq f(\mathbf{W}_{k-1}, \mathbf{H}_{k-1})$, where k denotes the iteration counter and f denotes the cost function of the minvol NMF problem. As f is bounded below, the sequence $\{f(\mathbf{W}_k, \mathbf{H}_k)\}_{k\in\mathbb{N}}$ converges.

Formal analysis using BSUM For the algorithms using the procedure mentioned in the previous subsections on $\mathcal{V}_{\text{det}}$ and $\mathcal{V}_{\text{logdet}}$, theoretical convergence can be guaranteed by the unified framework called *Block Successive Minimization* (BSUM) [93, Theorem 2].

The algorithm for solving minvol NMF with $\mathcal{V}_{\text{det}}$ is just a BCD, hence the convergence of this algorithm is covered by the theory of BCD, which is generalized by BSUM. Therefore, here we only focus on the convergence theory of BSUM, and in particular we focus on applying BSUM on the majorization-minimization algorithm of solving minvol NMF with $\mathcal{V}_{\text{logdet}}$.

We first state the conclusion of the theory. The theory of BSUM guarantees a sub-sequence of $\{\mathbf{W}_k, \mathbf{H}_k\}_{k\in\mathbb{N}}$ converges to a stationary point, under a set of assumptions listed in Table 2.2 , in which we explain the notation used in that table in the following paragraphs. Note that, to avoid making this subsection unnecessarily long, the assumptions in the table is not written as precise as they should be in the original form. The purpose here is to highlight the core idea of the theory of BSUM and to use it for explaining the convergence of minvol NMF; for a detailed treatment of these assumptions, see [93] .

We now briefly describe the framework of BSUM, and describe how the algorithm for solving minvol NMF with $\mathcal{V}_{\text{logdet}}$ mentioned in §2.2.2 fits into this framework. The problem we want to solve is to minimize, with respect to the variable $\mathbf{x}$, the function $f(\mathbf{x})$, subject to the constraint $\mathbf{x} \in \mathcal{X}$, where the set $\mathcal{X}$ can be expressed as the Cartesian product of n closed convex sets as $\mathcal{X} = \mathcal{X}_1 \times \cdots \mathcal{X}_n$. The variable $\mathbf{x}$ is split into n blocks of variable $x_1, \ldots, x_n$, where each block variable x_i carries the constraint $x_i \in \mathcal{X}_i$. This setting is satisfied in minvol NMF, where we have the block as $\mathbf{W}$ and $\mathbf{H}$, and here the sets $\mathcal{X}_i$ are just the nonnegative orthant.

In BSUM, we update only a single block in each iteration. Say, at iteration k, the variable block x_i is computed by solving the subproblem

$$\min_{x_i} u_i(x_i, x_{-i}) \quad \text{s.t.} \quad x_i \in \mathcal{X}_i, \tag{2.25}$$

where $u_i(\cdot,\cdot)$ is an upper bound function of the cost function f, and x_{-i} denotes the variable discarding the block x_i. This update corresponds to the update (2.22) in the majorization-minimization used in minvol NMF with $\mathcal{V}_{\text{logdet}}$: we do not minimize the cost function directly, instead we minimize an upper bound.

The key ingredient of BSUM is the function $u_i(\cdot,\cdot)$, which is a function of two arguments. Let y to be an arbitrary variable, then in order for the iterates of BSUM to converges, the function $u_i(\cdot,\cdot)$ has to satisfy all the assumptions listed in Table 2.2 , which we are going to explain one by one.

Table 2.2: The assumptions in the theory of BSUM [93, Theorem 2].

(A1)	$u_i(x_i, y)$ is quasi-convex in x_i.
(A2)	$u_i(y_i, y) = f(y)$.
(A3)	$u_i'(x_i, y, d_i)\|_{x_i=y_i} = f'(y, d_i)$.
(A4)	$u_i(x_i, y) \geq f(x_i, y_{-i})$.
(A5)	$u_i(x_i, y)$ is continuous in (x_i, y).
(A6)	The subproblem (2.25) has an unique solution.

Assumption (A1) is a standard assumption in the convergence analysis of BCD-type method [103], where the quasi-convexity is used to limit the number of minimum. Precisely, if u_i is quasi-convex, then u_i has at most one minimum in x_i. This assumption is satisfied for the majorizer of logdet presented in Lemma (2.2.1).

For assumptions (A2) to (A4), they mean u_i is the global majorizer of f (A4), and u_i has to "touch" the original cost function f at the point y (A2 and A3), meaning that the first-order behavior of u_i is the same as f locally. Together assumptions (A2) and (A4) also mean that u_i is a tight upper bound of f. These two assumptions are needed as they "connect" the minimization of f and minimization of u. These assumptions are satisfied for the majorizer of logdet presented in Lemma (2.2.1).

Assumptions (A5) and (A6) are again standard assumptions in the convergence analysis of BCD-type method. These assumptions are satisfied for the majorizer of logdet presented in Lemma (2.2.1).

After we explained Table 2.2 , we restate again the conclusion of the theory of BSUM: when the assumptions (A1)-(A6) are satisfied, then the theory of BSUM guarantees the sub-sequence of $\{x_{ik}\}_{k\in\mathbb{N}}$ converges to a stationary point. As the majorizer of logdet presented in Lemma (2.2.1) satisfies these assumptions, hence the theory of BSUM applies to the algorithm for the minvol NMF with $\mathcal{V}_{\text{logdet}}$.

On algorithm for Minvol NMF with Nuclear norm volume The approach for solving subproblem on $\mathbf{W}$ with $\mathcal{V}_*$ is a heuristic, and there is no theoretical convergence guarantee. However, it is observed that the sequence converges empirically in the hyperspectral imaging application to be presented in the next chapter.

2.3 Chapter summary and perspectives

Chapter summary We introduced the minvol NMF. We investigated minvol NMF models with different volume regularizations were investigated, in which we talked about the uses of different volume regularizers, the relationships between these regularizers and some comparisons in the practical setting such as noisy cases and rank deficient cases. We provide an identifiability result on a particular minvol NMF model, namely minvol NMF using determinant volume, and show that the solution is provably unique when the SSC assumption is satisfied. Finally, we briefly discuss how to solve the minvol NMF problems.

Open problems We now state the opening problems concerning the minvol NMF.

- **Automatic tuning for δ for $\mathcal{V}_{\textbf{logdet}}$** In §2.1.3 we discussed the issue of selecting the parameter δ for the minvol NMF model using regularizer $\mathcal{V}_{\text{logdet}}$. Adapting the automatic tuning strategy for δ in the context of matrix completion to the context of minvol NMF will be a interesting topic of further research.

- **Robustness of minvol NMF for the $\mathcal{V}_{\textbf{det}}$ and $\mathcal{V}_{\textbf{logdet}}$** Unlike SNMF in which there are some robustness results, currently there is no theoretical robustness analysis on minvol NMF model (2.1a). This is a promising but difficult direction of further research.

- **Identifiability of minvol NMF using $\mathcal{V}_{\textbf{logdet}}$, $\mathcal{V}_*$ and $\mathcal{V}_{p,\delta}$** The identifiability result developed so far (say, Theorem 2.1.1, and those in [77, 42, 39]) are all based on using the SSC condition (Definition (2.1.3)). Such proof technique only works for the det regularizers, but not for the logdet function, Nuclear norm, Frobenius norm and the smooth Schatten p-norm. How to generalize the identifiability result to minvol models with other regularizers remains open.

- **Theoretical analysis of the automatic model order selection functionality of logdet volume** As we stated in §2.1.3 that, logdet regularizer works even in rank deficient case, and it has the ability of automatic model order selection, which will be shown in §4.2 when using minvol NMF on audio data. Developing the theoretical understanding of this behavior of the logdet regularizer is an interesting research problem.

- **Effective computation for solving minvol NMF with $\mathcal{V}_{\textbf{det}}$ and $\mathcal{V}_*$** When solving the subproblem on $\mathbf{W}$ for minvol NMF with $\mathcal{V}_{\text{det}}$, the solution approach relies on solving the QP subproblem (2.18), in which the coefficients in the QP can be expensive to compute. This makes the iterative algorithm for solving minvol NMF with $\mathcal{V}_{\text{det}}$ expensive. Therefore, it is helpful to develop more efficient algorithms for solving such minvol NMF problem. For the approach to solve minvol NMF with $\mathcal{V}_*$, the current method is just a heuristic with no theoretical convergence guarantee, so an algorithm with theoretical support is a direction of future research.

3 Minvol NMF on hyperspectral unmixing

> Colour as perceived by us is a function of three independent variables.
>
> *James Clerk Maxwell*

In this chapter, we discuss NMF applied to hyperspectral unmixing.

Chapter organization We first give a brief introduction about Hyperspectral Unmixing (HU) in §3.1 , we discuss the data format and the goals of HU. Then in §3.2 we discuss how the algorithms are compared. Finally in §3.3 we present the numerical results on comparing NMF algorithms presented in §2 with some other algorithms.

Highlights of contributions This chapter is to showcase the effectiveness of minvol NMF on the HU application. Here the contribution is the empirical results listed in §3.3 on using minvol NMF to solve HU problems. The experimental results showed that minvol NMF algorithms are superior to other algorithms in HU problems on both synthetic and real datasets. In particular, the minvol NMF algorithms outperform both the state-of-the-art separable NMF algorithm `SPA`, and two volume-based methods, namely `MVC-NMF` and `RVolMin`.

3.1 Hyperspectral unmixing

We now give a brief introduction to Hyperspectral Unmixing (HU); see [81, 12] and the references therein for more discussions.

Hyperspectral image The goal of (blind) HU is to study the composition and the distribution of materials in a given scene being imaged. A scene usually consists of a few fundamental types of materials called *endmembers*. In HU, the goal is to get the information of these endmembers from the observed Hyper-Spectral Image (HSI), which is captured by sensors over different wavelengths in the electromagnetic spectrum. These images form a 3rd-order data tensor of size $m \times \text{col} \times \text{row}$, with m is the number of spectral bands, "col" and "row" are the dimensions of the images. With $n = \text{col} \times \text{row}$, the m-by-n data matrix $\mathbf{M}$ is obtained by stacking the m-dimensional spectral signature of the pixels as the rows of $\mathbf{M}$. See Fig.1.2 and Fig. 3.1 for illustrations.

Example: the Jasper Ridge image Fig. 3.1 shows a HSI scene of the Upper Crystal Springs Reservoir[1] of the Jasper Ridge data and the Google Maps satellite image (not taken in the same date as the HSI data). The scene consists mostly of four basic endmembers: water, vegetation (tree and grass), dirt (soil) and road.

The HU problem Table 3.1 lists the typical goals of blind HU on a HSI data. The first problem is model order selection, recall that model order selection is a topic of research on its own, therefore, we follow what we stated in §1.4 , we assume r is given. The main focus of HU here is the second problem. In fact, assuming (2) in Table 3.1 is solved, then (3) in Table 3.1 can be tackled by solving a NNLS.

[1] Located in California, United States. Google Maps coordinate: @37.4950658,-122.3303297,4733m.

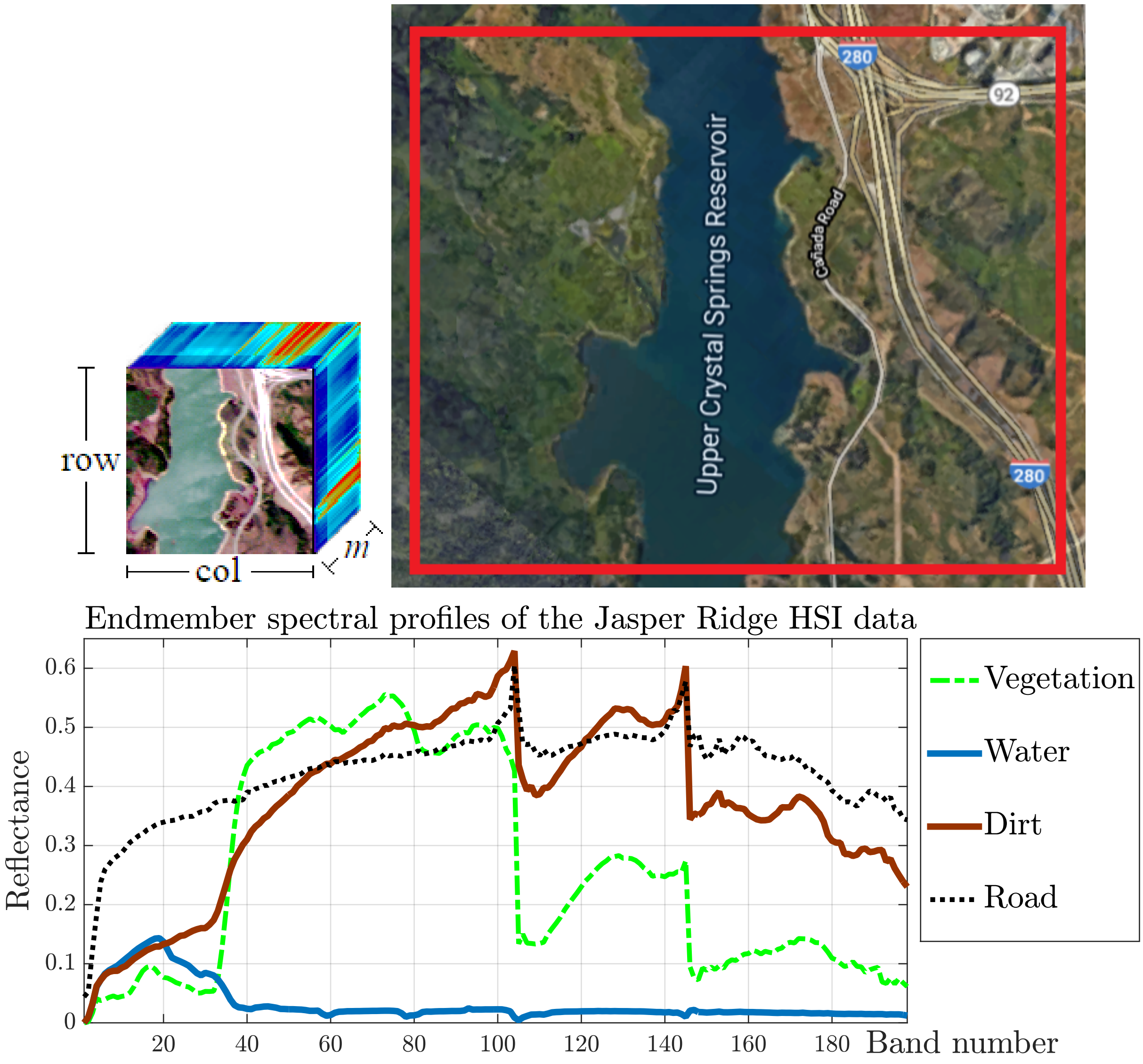


Fig. 3.1. The hyperspectral image of Jasper Ridge and the spectral profiles of the four endmembers. **Top**: The hyperspectral image of Jasper Ridge and the image in Google Maps. **Bottom**: The spectral profiles of the four endmembers.

Linear spectral unmixing We now explain the correspondence between HU and NMF in Table 3.1. A widely used model for HU is the (low-rank) linear mixing model:

$$\mathbf{M} = \mathbf{W}^{\text{true}}\mathbf{H}^{\text{true}} + \mathbf{N},$$

where the data $\mathbf{M}$ is generated by $\mathbf{W}^{\text{true}}$ (the basis, where each column of $\mathbf{W}^{\text{true}}$ is the spectral signature of an endmember), weighted by the matrix $\mathbf{H}^{\text{true}}$ plus noise $\mathbf{N}$. The matrix $\mathbf{H}$ in HU is called the *abundance matrix* and encodes how much each endmember (columns of $\mathbf{W}^{\text{true}}$) is presented in each pixel of the image: $H_{k,j}$ is the abundance of the kth endmember in the jth pixel. There are three physical constraints in HU: the nonnegativity constraints $\mathbf{W} \geq 0$ (spectral signatures are nonnegative), $\mathbf{H} \geq 0$ (abundances are nonnegative) and the sum-to-one constraint $\mathbf{H}^\top \mathbf{1}_r = \mathbf{1}_n$ (abundances in each pixel sum to one). We consider a more general constraint for $\mathbf{H}$, namely using $\mathbf{H}^\top \mathbf{1}_r \leq \mathbf{1}_n$, that allows to take into account different intensities of illumination among the pixels of the image. So in terms of computational point of view, we should also consider the constraint $\mathbf{H}^\top \mathbf{1}_r \leq \mathbf{1}_n$. Now we see NMF (in particular, the minvol NMF model (2.1a)) is the right model to perform blind HU under the linear mixing model, i.e., to learn the endmember matrix $\mathbf{W}$ and the

Table 3.1: Three typical goals in HU.

	HU objective	Task in NMF
(1)	Identify the number of endmembers.	Find r.
(2)	Obtain the spectral profile of the endmembers.	Find $\mathbf{W}$.
(3)	Identify which pixel contains which endmember and in which proportion.	Find $\mathbf{H}$.

abundance matrix $\mathbf{H}$ from the data. We recall that a pictorial description on how NMF is used in HU is given in Fig.1.2 in §1.4.2 .

About the endmember Endmembers are the macroscopically pure components that are assumed to be homogeneously distributed in a pixel. For example, in Fig. 3.1 , we assume different pixels containing "water" are identical, and, we do not consider differentiating different type of water (e.g. deep water vs shallow water). Another example is the vegetation. Part from tree and grass are different type of vegetation, the amount of chlorophyll and the amount of dry matters in the trees can be different, all of these results in a phenomenon known as Spectral Variability on the endmember spectrum. That is, the real spectral profile of different trees are not completely identical, but in the macroscopic endmember model, they are assumed to be the same.

Nonlinear unmixing It is possible to use nonlinear mixing model for HU, which can uncover the microscopically pure components in the data. For example, we can follows the idea of *Kernel Trick* and take the model as $\mathbf{M} = \Phi(\mathbf{W}^{\text{true}})\mathbf{H}^{\text{true}} + \mathbf{N}$, where Φ is a nonlinear mapping. Or, we can consider quadratic mixing model as $\mathbf{M} = \mathbf{W}^{\text{true}}\mathbf{H}_1^{\text{true}} + (\mathbf{W}^{\text{true}} \circ \mathbf{W}^{\text{true}})\mathbf{H}_2^{\text{true}} + \mathbf{N}$ for $\circ$ denoting second order interactions between columns of $\mathbf{W}^{\text{true}}$. These models are out of the scope of the thesis.

Separability in HU: the pure-pixel assumption Recall that we discussed Separable NMF (SNMF) and the separability condition in §1.5 . In fact, SNMF solves blind HU problems when the data satisfy the separability condition, which is also known as the *pure-pixel assumption* in HU. It means that the data contain at least r pure pixels where each pure pixel contains only one endmember, and there is a one-to-one correspondence between the r pure pixels and the r endmembers. In the example of Fig. 3.1 , there are regions in the image containing pure components, it appears that the dataset is likely to satisfy separability.

As discussed in §1.5 that SNMF is geometrically a vertex identification problem: given a noiseless $\mathbf{M}$, locate the extreme points $\mathbf{W} = \mathbf{M}(:, \mathcal{A})$ which will be exactly $\mathbf{W}^{\text{true}}$ if the separability condition holds. Many algorithms exist to perform this task (referred to as pure-pixel search algorithms in HU), such as the `SPA` (§1.5.1). For other algorithms, see [45].

We now illustrate the use of NMF in HU. Now the data has no water region, it contains 3 endmembers: vegetation, dirt and road. So we run `SPA` on this part of the data with $r = 3$, the result is shown in Fig.3.2 .

Quantifying the violation of separability When the separability condition does not hold, as illustrated in the example in Fig.3.2 , pure-pixel search algorithms fail and we need to switch to minvol algorithms. We quantify how much separability is violated by introducing the *non-separability pa-*

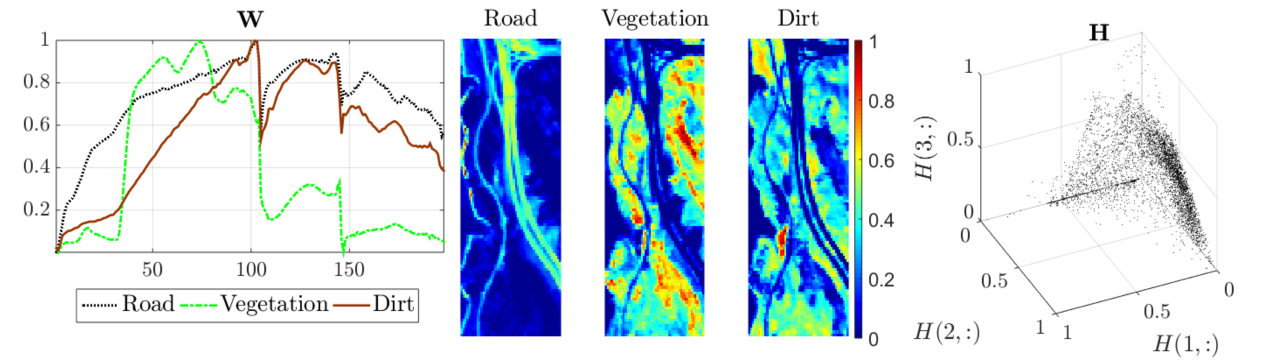


Fig. 3.2. The result from SPA on part of the Jasper Ridge data: the right-half part of the image data in in Fig. 3.1 . Here **W** are close to the one in Fig. 3.1 , the y-axis of **W** in this figure is different from that in Fig. 3.1 due to scaling ambiguity. Three abundance maps, obtained by reshaping rows of **H** back to "col" × "row", are plotted. These abundance maps correspond to the distribution of the material as shown in Fig. 3.1 . We also plotted the coordinates of **H**, we can see that the points in **H** are wide spread in component 2 and 3 but not for component 1, therefore this data is in fact not fully separable.

rameter $\mathbf{p}$, which is a r-dimensional vector in $[0,1]^r$. First, note that separability holds if each row of $\mathbf{H}$ contains at least one entry equal to one, that is, $\|\mathbf{h}^j\|_\infty = 1$ for $j \in [r]$. So, to break separability, we need $\|\mathbf{h}^j\|_\infty \leq p_j < 1$ for some j. Fig. 3.3 gives an example with $r = 4$. Hence, having $\|\mathbf{h}^j\|_\infty \leq p_j$ means that the maximum abundance of the jth endmember in all pixels is at most p_j, which sets a minimal amount of separation between the data points (the black dots in Fig. 3.3) to the vertex $\mathbf{w}_j$ (the black stars). In other words, p_j controls the gap between $\{\mathbf{x}_i\}_{i\in[n]}$ and $\mathbf{w}_j$, where $\mathbf{x}_i$ is the ith data point (ith column of $\mathbf{X}$). Note that the entries of p can be different, which gives *asymmetric non-separability*. For example, in Fig. 3.3 , data points are closer to vertex 1 than vertex 4.

minvol NMF in HU In Fig. 3.3, the reconstructions given by `SPA` (green vertices) and `RVolMin` [41] (a state-of-the-art minimum-volume algorithm using $\mathcal{V}_{\text{logdet}}$; deep blue vertices) are far away from the ground truth, and minvol NMF with $\mathcal{V}_{\text{det}}$ (cyan vertices) produced perfect recovery (black vertices). This is supported by Theorem 2.1.1, which is the theoretical ground of using minvol NMF for solving HU problems. Recall that, in the absence of pure pixels, Theorem 2.1.1 guaranteed the recovery of the endmember when solving minvol NMF, subject to the SSC condition.

3.2 Common performance metrics for ranking algorithms

In this section we discuss the common setting in comparing the performance of NMF algorithms.

The structure of a generic algorithm A general (iterative) algorithm $\mathcal{A}$ is in the form

$$\text{solution} \leftarrow \mathcal{A}(\text{input data}, \text{parameters}, \text{termination criterion}).$$

For NMF, the input is the data matrix $\mathbf{M}$. For parameters, the typical inputs are the factorization rank r and the regularization weights, if there is a regularization. Sometimes there are additional model-specific parameters, e.g., in the minvol NMF using $\mathcal{V}_{\text{logdet}}$, there is a δ parameter in the logdet term. For the termination criterion, an algorithm is stopped when it reached (i) a certain predefined

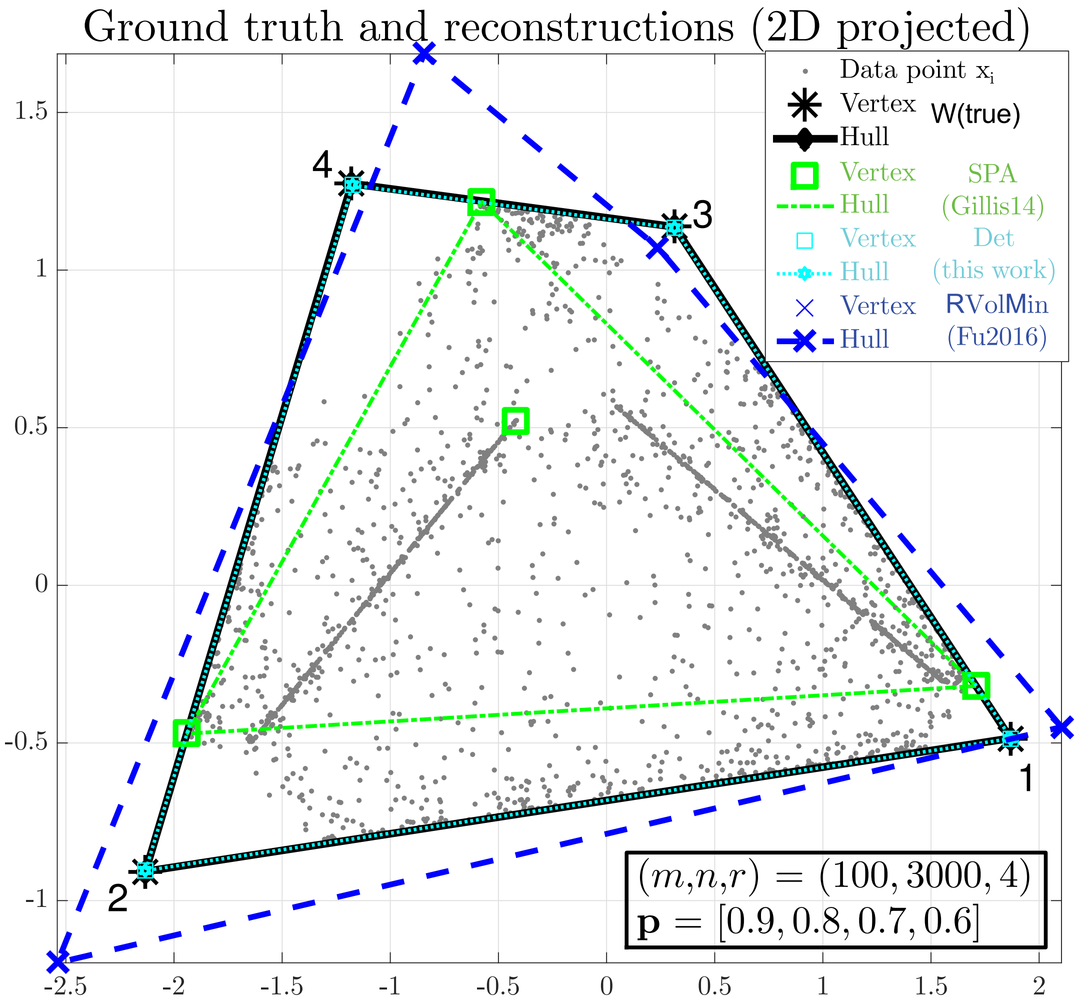


Fig. 3.3. A toy example of minvol NMF on synthetic data. The plot shows the projection of data points (black dots) onto a two-dimensional plane using PCA. This data set was generated with $\mathbf{p} = [0.9, 0.8, 0.7, 0.6]$, meaning the maximum abundance of each ground truth vertex ($\mathbf{W}^{\text{true}}$, black stars) in any pixel is at most 90% for vertex 1, 80% for vertex 2, 70% for vertex 3, and 60% for vertex 4. In the legend, "Det" refers to minvol NMF with $\mathcal{V}_{\text{det}}$. Figure copied from [6].

precision, or (ii) the maximum number of allowed iterations, or (iii) the maximum allowed runtime. In this thesis, we mostly focus on the last two termination criterion.

Finally, outputs are in general the factor matrices $\mathbf{W}$ and $\mathbf{H}$. Using these matrices we can define performance measures for comparing different algorithms. These performance measures are often quantified by the size of the error. Three commonly used error measures are discussed below.

Data fitting error This is the most simple type of error, which is defined as

$$\text{Data fitting error} := \|\mathbf{M} - \mathbf{WH}\|_F. \tag{3.1}$$

We also define relative data fitting error as

$$\text{Relative Data fitting error} := \frac{\|\mathbf{M} - \mathbf{WH}\|_F}{\|\mathbf{M}\|_F}. \tag{3.2}$$

Factor fitting error When the ground truth factor is available, we can rank the algorithms by measuring how close the solution they produced to the ground truth factor. For example, on $\mathbf{W}^{\text{true}}$, the factor fitting error is defined as

$$\text{Factor fitting error} := \|\mathbf{\Lambda W \Pi} - \mathbf{W}^{\text{true}}\|_F, \tag{3.3}$$

where $\mathbf{\Pi}$ is a permutation matrix to match the columns between $\mathbf{W}$ and $\mathbf{W}^{\text{true}}$, and $\mathbf{\Lambda}$ is positive diagonal matrices to properly scale the size of the columns in $\mathbf{W}$ to match that of $\mathbf{W}^{\text{true}}$. These are necessarily corrections for handling the permutation and scaling ambiguity (1.8), as direct subtraction $\mathbf{W} - \mathbf{W}^{\text{true}}$ often produces wrong result. The matrix $\mathbf{\Lambda}$ can be computed easily using any specific normalization. The computation of $\mathbf{\Pi}$ is much more difficult. In fact, usually $\mathbf{\Pi}$ it obtained by solving an *Assignment Problem*:

$$\underset{x_{ij}, i\in[n], j\in[m]}{\text{minimize}} \sum_{i,j} c_{ij}x_{ij} \text{ subject to } \sum_{i=1}^{n} x_{ij} = 1, \ \sum_{j=1}^{m} x_{ij} = 1 \text{ and } x_{ij} \in \{0,1\},$$

where $c_{ij} := -\left|\dfrac{\langle \mathbf{w}_i, \mathbf{w}_j^{\text{true}}\rangle}{\|\mathbf{w}_i\|_2\|\mathbf{w}_j^{\text{true}}\|_2}\right|$ is the cost of assigning column $\mathbf{w}_i$ to column $\mathbf{w}_j^{\text{true}}$. This combinatorial optimization problem can be solved efficiently by using the Munkres algorithm (also known as the Hungarian algorithm), or as a linear programming problem.

Mean Removed Spectral Angle Mean Removed Spectral Angle (MRSA) is another way to measure the factor fitting quality. MRSA between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m_+$ is defined as

$$\text{MRSA}(\mathbf{x}, \mathbf{y}) := \frac{100}{\pi} \cos^{-1}\left(\frac{\langle \mathbf{x} - \bar{\mathbf{x}}, \mathbf{y} - \bar{\mathbf{y}}\rangle}{\|\mathbf{x} - \bar{\mathbf{x}}\|_2\|\mathbf{y} - \bar{\mathbf{y}}\|_2}\right) \in [0, 100]. \tag{3.4}$$

where $\bar{\mathbf{x}}$ denotes the mean of entries of $\mathbf{x}$. In other words, MRSA is the cosine-distance between $\mathbf{x}$ and $\mathbf{y}$, after removing the effects of shifting and scaling. A low MRSA value means a good matching between $\mathbf{x}$ and $\mathbf{y}$.

For MRSA between $\{\mathbf{w}_i\}_{i\in[r]}$ and $\{\mathbf{w}_i^{\text{true}}\}_{i\in[r]}$, it can be defined as the mean of MRSA between each vector pair $\{\mathbf{w}_i, \mathbf{w}_i^{\text{true}}\}$, after we matched the columns $\{\mathbf{w}_i\}_{i\in[r]}$ and $\{\mathbf{w}_i^{\text{true}}\}_{i\in[r]}$ by solving the assignment problem mentioned above. Mathematically,

$$\text{MRSA}(\mathbf{W}, \mathbf{W}^{\text{true}}) := \frac{100}{\pi r} \sum_{i=1}^{r} \cos^{-1}\left(\frac{\langle \mathbf{w}_i - \bar{\mathbf{w}}_i, \mathbf{w}_i^{\text{true}} - \bar{\mathbf{w}}_i^{\text{true}}\rangle}{\|\mathbf{w}_i - \bar{\mathbf{w}}_i\|_2\|\mathbf{w}_i^{\text{true}} - \bar{\mathbf{w}}_i^{\text{true}}\|_2}\right) \in [0, 100]. \tag{3.5}$$

Comparing factor fitting error and MRSA Both factor fitting error (3.3) and MRSA (3.5) quantify how close $\mathbf{W}$ is to $\mathbf{W}^{\text{true}}$. In HU application, MRSA gives a better measurement than factor fitting error since it purely depends on the “shapes” of the vectors.

3.3 Experiments on using NMF to solve HU problems

In this section we present the numerical results on using minvol NMF to solve HU problems. This section is organized as follows. First we describe the experiment setup, then we report the results on semi-synthetic data on the task of recovering the ground truth $\mathbf{W}^{\text{true}}$ under different asymmetric non-separability and noise levels. Finally, we present results on two real-world datasets.

3.3.1 Experimental set up

Data generation In the (semi-)synthetic experiments, we generate the observed data matrix as $\mathbf{X} = [\mathbf{X}^{\text{clean}} + \mathbf{N}]_+$ with white Gaussian noise $\mathbf{N} \in \mathbb{R}^{m\times n}$ with zero mean and variance $\sigma \geq 0$, where

$[\cdot]_+$ denotes nonnegative projection. Here $\mathbf{X}^{\text{clean}} = \mathbf{W}^{\text{true}}\mathbf{H}^{\text{true}}$, where $\mathbf{W}^{\text{true}}$ comes from datasets from [115][2], see Fig. 3.1 and Fig. 3.4. We generate each column of $\mathbf{H} \in \mathbb{R}_+^{r \times n}$ with $n = 1000$ using the Dirichlet distribution of parameter 0.1, which is a reasonable choice for HU data.

Now we talk about the non-separability parameter $\mathbf{p}$. If $\mathbf{p} = \mathbf{1}_r$, i.e., separability condition holds, we take $\mathbf{H}$ as $\mathbf{H}^{\text{true}}$. Otherwise, we remove those columns of $\mathbf{H}$, denoted as $\mathbf{h}_\#$ where $\mathbf{h}_\#(j) > p_j$, and re-sample again until $\mathbf{H}$ satisfies the condition $\|\mathbf{h}^j\|_\infty \leq p_j$ for all j. We use $p_j \in (0.5, 0.99]$ for all j.

For experiments on real data, we use the datasets in Fig.3.1 and Fig.3.4 untouched, that is, with no preprocessing. We emphasize that the goal here is to demonstrate NMF can be used to solve HU problem. It is important to note that, preprocessing itself plays a very important role in data analysis, which can greatly influence the result. In HU, preprocessing includes cloud removal, fog removal, intensity correction, denoising and outlier removal. In our experiment, we perform no dimension reduction and no preprocessing.

Finally, we set the maximum number of iterations to 300.

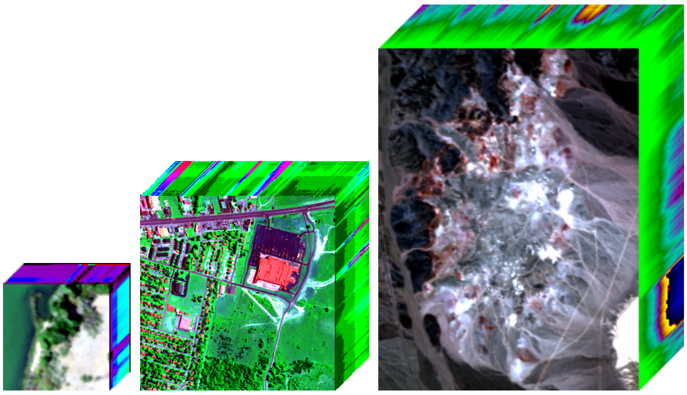


Fig. 3.4. HSI datasets. From left to right: "Samson" $(m, r) = (156, 3)$, "Urban" $(m, r) = (162, 6)$, and "Cuprite" $(m, r) = (224, 12)$. Figure modified from [6]. See [6] for the endmembers of the datasets.

Parameters of the algorithms First, we name the algorithm solving the minvol NMF using the name of the volume regularizer used in the model. e.g., we name the algorithm solving minvol NMF using $\mathcal{V}_{\text{det}}$ as `Det`. Similarly, we have `logdet` for minvol NMF with $\mathcal{V}_{\text{logdet}}$ and `Nuclear` for minvol NMF with $\mathcal{V}_*$. For how to solve minvol NMF problems, see §2.2 . We compare `logdet`, `det` and `Nuclear` with `SPA` and `RVolMin` [41], a state-of-the-art volume-regularized method. For `RVolMin`, we use the same data fitting term (namely, the Frobenius norm), the same number of iterations, the same regularization parameter search scheme and the same initialization as for the minvol NMF, however, we follows [41] on using $\delta = 10^{-8}$ for `RVolMin` that uses $\mathcal{V}_{\text{logdet}}$.

Given an observed data matrix, all minvol NMF algorithms have two main parameters: r and λ, they also require an initialization $(\mathbf{W}^{\text{ini}}, \mathbf{H}^{\text{ini}})$. As stated in §1.4 that we assume r is known, we just use the r values as shown in Fig.3.1 and Fig.3.4 . Next, we generate $\mathbf{W}^{\text{ini}}$ from data using `SPA`, and generate $\mathbf{H}^{\text{ini}}$ using the method stated in §2.2.1 .

Tuning of regularization parameter For the regularization parameter λ, as discussed in §2.1.3, it should usually be chosen small. In fact, a large λ forces the vertices of the convex hull of $\mathbf{W}$ to

[2] `http://lesun.weebly.com/hyperspectral-data-set.html`

be very close to each other, making $\mathbf{W}$ ill-conditioned or rank-deficient. In particular, for minvol NMF with $\mathcal{V}_*$, the SVT operator set singular values smaller than λ to zero, so a large λ makes $\mathbf{W}$ rank-deficient, see §2.2.2 for details.

The following describes how we tune λ. The goal here is to tune λ so that each algorithm performs as best as possible for the considered problems. Here we use the ground truth $\mathbf{W}^{\text{true}}$ to show case the effectiveness of the bisection. We set $\lambda = \tilde{\lambda}\frac{f(\mathbf{W}^{\text{ini}},\mathbf{H}^{\text{ini}})}{|\mathcal{V}(\mathbf{W}^{\text{ini}})|}$, where the initialization pair $(\mathbf{W}^{\text{ini}}, \mathbf{H}^{\text{ini}})$ is provided by `SPA`, and $\tilde{\lambda}$ is a tuning variable within the search interval $\mathcal{I}_0 = [10^{-6}, 0.5]$. We run greedy bisection search to tune $\tilde{\lambda}$:

- (step-1) Take $\tilde{\lambda}_a = 10^{-6}$, $\tilde{\lambda}_b = 0.5$ initially.

- (step 2) Run the algorithm with $\tilde{\lambda}_a$, $\tilde{\lambda}_b$ and $\tilde{\lambda}_c$ where $\tilde{\lambda}_c = (\tilde{\lambda}_a + \tilde{\lambda}_b)/2$. Denote the performance of the solution $\mathbf{W}$ produced under a specific $\tilde{\lambda}$ as MRSA$(\tilde{\lambda})$, see §3.2 for the definition of MRSA.

- (step-3) Split the search interval $\mathcal{I}_0 = [\tilde{\lambda}_a,\tilde{\lambda}_b]$ into two intervals $\mathcal{I}_0^1 = [\tilde{\lambda}_a, \tilde{\lambda}_c]$ and $\mathcal{I}_0^2 = [\tilde{\lambda}_c, \tilde{\lambda}_b]$. For each interval there are two MRSA values, we let the MRSA value of an interval be the sum of these values.

- (step-4) We repeat step-1 to step-3 on the interval with the lowest MRSA value. i.e., at iteration k, we shrink the search interval by half by defining the new search interval $\mathcal{I}_{k+1}$ as
$$\mathcal{I}_{k+1} \leftarrow \min_{\mathcal{I}} \Big\{ \text{MRSA}(\mathcal{I}_k^1), \text{MRSA}(\mathcal{I}_k^2) \Big\}.$$

- If a draw happens in step-4, we perform two bisections on each of the interval $\mathcal{I}_k^1$ and $\mathcal{I}_k^2$ and set the next interval to be the one with the smallest MRSA.

- We repeat this process (step-1 to step-4) at most 20 times, or if the improvement from one iteration to the next is negligible, namely
$$\left|\text{MRSA}\left(\tilde{\lambda}_{k+1}\right) - \text{MRSA}\left(\tilde{\lambda}_k\right)\right| \leq 10^{-4}.$$

Note that

- Such greedy bisection search does not guarantee $\tilde{\lambda}_k$ to converge to the best value, i.e., the value that corresponds to the lowest MRSA.

- In real application we often do not have access to the ground truth $\mathbf{W}^{\text{true}}$. However we can still perform the bisection to tune λ based on inspecting the solution (say, judging on how well the data is, or how the abundance map fit to the actual scene, and how the spectrum looks like).

Extensive numerical experiments showed the effectiveness of this scheme, which will be illustrated in the next paragraph.

Effectiveness of the bisection search on tuning regularization parameter Here we illustrate the effectiveness of the tuning strategy for λ. We perform experiments on the synthetic dataset where $\mathbf{W}^{\text{true}}$ comes from the Samson dataset with $r = 3$ as follows. We consider 6 different values of the symmetric non-separability vector $\mathbf{p} = [p_1, p_2, p_3]$ where $p_1 = p_2 = p_3$ are selected from the set $\{0.93, 0.89, 0.86, 0.83, 0.79, 0.76\}$. We solve the minvol NMF with $\mathcal{V}_{\text{det}}$ with two parameters tuning schemes:

1. the bisection search mentioned in 3.3.1 , and

2. a brute-force grid search: we solve the minvol NMF with all 100 equally spaced steps values of $\tilde{\lambda}$ in the interval $\mathcal{I}_0 = [10^{-6}, 0.5]$.

For each value of $\mathbf{p}$, 10 datasets are generated randomly. We denote λ^{bi} the value of λ obtained by the bisection search and λ^{grid} by the grid search. Table 3.2 shows the results between comparing the bisection search and the grid search displayed as $\Big(\text{MRSA}(\lambda^{\text{bi}}) - \text{MRSA}(\lambda^{\text{grid}})\Big)/|\text{MRSA}(\lambda^{\text{grid}})|$ in the format of (mean±std) and the average number of iterations performed by the bisection search to reach λ^{bi}.

Table 3.2: Comparison of MRSA values of the bisection and the grid search to obtain λ^{bi} and λ^{grid}, respectively. The second column is the number of iterations needed for the bisection search to terminate. The negative values in the first column indicate the bisection search performs better than the grid search.

p	$\Big(\text{MRSA}(\lambda^{\text{bi}}) - \text{MRSA}(\lambda^{\text{grid}})\Big)/|\text{MRSA}(\lambda^{\text{grid}})|$	#iterations
0.93	-0.0016±0.0002	2±0.00
0.89	-0.0143±0.0023	6.4±0.52
0.86	-0.0235±0.0029	7.9±0.32
0.83	-0.0429±0.0032	9±0.00
0.79	**-0.0815±0.0089**	11.1±0.32
0.76	-0.0419±0.0032	10.5±0.53

From Table 3.2 , we make the following two interesting observations:

1. As the purity goes down, more bisections are need to identify the best λ. This was expected since, for a high purity, the initialization (`SPA`) provides a good initial solution.

2. In all 50 cases, the value of λ^{bi} lead to a slightly smaller MRSA value than λ^{grid} while requiring less computations. This illustrates the fact that the bisection is able to identify the right value of λ and to refine the search around that value with more precision than an expensive exhaustive search. In fact, by comparing to the 100 search iterations in the brute-force approach, we can see a reduction in the number of search in the bisection approach.

3.3.2 Numerical results on HU

Preliminary experiments on comparing with MVC-NMF `MVC-NMF` is one of the first minvol NMF type of algorithm[3] [84]. We run a small experiment to showcase `MVC-NMF` does not compete with minvol NMF algorithms and `RVolMin` [41]. Similarly as in the previous section, we use synthetic datasets where $\mathbf{W}^{\text{true}}$ comes from the Samson dataset with $r = 3$. However we test in a more difficult scenarios where the non-separability vectors $\mathbf{p} = [p_1, p_2, p_3]$ has p_1 selected from $\{0.95, 0.9, 0.85, 0.8, 0.75\}$ with $p_2 = 0.79$ and $p_3 = 0.69$. For each value of $\mathbf{p}$, 10 datasets are generated randomly. All algorithms take the same initialization (`SPA`), the same number of iterations (100) and the regularization parameter λ is tuned using the bisection search.

[3] Code available at `https://github.com/aicip/MVCNMF`

Fig. 3.5 shows that MVC-NMF produces significantly worse results than others, it will not be considered in the experiments in the subsequent comparisons. It is difficult to pinpoint the reasons of the poor results of MVC-NMF; we see at least two of them: 1) MVC-NMF uses another regularizer which is the squared volume of the convex hull of the columns of $\mathbf{W}$ without the origin. 2) MVC-NMF does not make use of advanced optimal optimization method. For example it does not make use of any acceleration technique and the convergence can be slow.

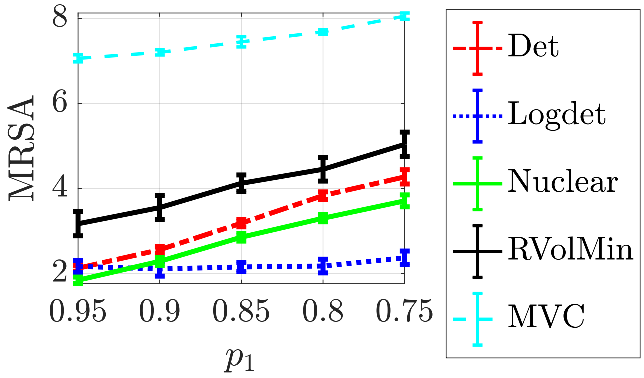


Fig. 3.5. MRSA curves of minvol NMF with different $\mathcal{V}$, RVolMin and MVC across different values of p_1 ($p_2 = 0.79$, $p_3 = 0.69$) on synthetic Samson datasets.

Now we run more comprehensive experiments on the synthetic datasets.

On Samson dataset with r = 3 Here we run $10 \times 10 \times 10 = 1000$ experiments where the data in each experiment are generated using the non-separability vector $\mathbf{p} = [p_1, p_2, p_3]$, where p_i is selected from the set

$$\mathbb{P} = \{0.99, 0.96, 0.93, 0.89, 0.86, 0.83, 0.79, 0.76, 0.73, 0.69\}.$$

Fig. 3.7 shows the results in form of MRSA cubes for the algorithms SPA, Det, logdet, Nuclear and RVolMin, where each pixel in the cube is the result (in MRSA) over one trial. We then construct the recovery curves by counting the number of cases (pixels) in the cube that the MRSA value is less than a threshold; see Fig. 3.6. In general, the results in Fig. 3.7 and 3.6 show that

- All minvol NMF algorithms perform better than the state-of-the-art SNMF algorithm SPA, as the MRSA cubes of minvol NMF have a wider blue region (lower MRSA) and their recovery curves in Fig. 3.6 dominates that of SPA.

- When the data is highly non-separable (low p_i's), minvol NMF algorithms perform worse than SPA. In fact, the region of the cube corresponding to highly non-separable data in SPA is not as red as for the minvol NMF approaches. The reason is that SPA always extract points from the data hence these points are never too far from the vertices. However, when the data is highly non-separable, minvol NMF may generate points further away than the vertices.

- Compared with RVolMin, logdet and Nuclear are consistently better, while Det performs similarly.

- logdet performs better than Det and Nuclear for highly non-separable data, as its red region is much more concentrated around the highly non-separable corner of the cube.

In terms of computational time, in seconds, Det takes 2.1±0.2, logdet takes 1.2±0.1, Nuclear takes 1.3±0.2, and RVolMin takes 2.1±0.0. It is expected Det to take longer run time than other algorithms as the computation of the coefficients of the QP in Det is expensive, see QP (2.21) and the discussion therein.

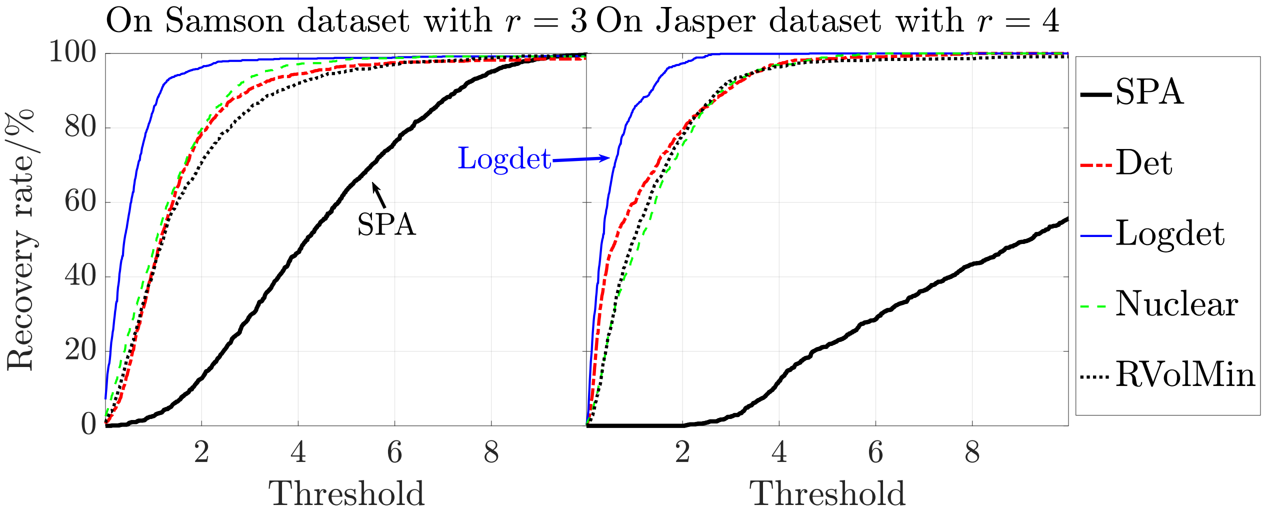


Fig. 3.6. Curves of recovery. **Left**: Corresponds to Fig. 3.7 for the synthetic Samson datasets with $r = 3$. **Right**: Corresponds for the dataset Jasper with $r = 4$, with the same experimental set up as the one in Fig. 3.7 , where p_4 fixed at 0.75.

Using the Jasper dataset with r = 4 In Fig. 3.6 , we run the same experiment on the dataset Jasper with $r = 4$, where we fix $p_4 = 0.75$. Furthermore, Table 3.3 shows the result in MRSA on the dataset Jasper across three sets of predefined $\mathbf{p}$ values (highly separable with $\mathbf{p}_{\text{high}} = [0.9, 0.8, 0.7, 0.6]$, less separable with $\mathbf{p}_{\text{mid}} = [0.8, 0.7, 0.6, 0.51]$, and even less separable with $\mathbf{p}_{\text{low}} = [0.7, 0.65, 0.55, 0.51]$) and three noise levels ($\sigma = 0.001, 0.005$ and 0.01). Each item in the table (in mean±std) is the average over 20 trails. We observe the following

- Fig. 3.6 shows that minvol NMF algorithms are competitive with the state-of-the-art minimum volume based method RVolMin, where Det performs slightly better than RVolMin while logdet is significantly better.

- Tables 3.3 shows that all the methods improve the fitting accuracy of SPA, with Det has the best performance in all cases and RVolMin has the worst performances in all cases. In the table, the best results are bolded, and the red color denotes the worst result (excluding SPA as it is used for initialization). The same table convention will be used in the whole thesis.

In terms of computational time, in seconds, Det takes 6.62±0.5, logdet takes 2.22±0.2, Nuclear takes 1.45±0.0 and RVolMin takes 2.7±0.0. For the same reasons as stated in the previous experiment, Det is slower.

Using the Urban dataset with r = 6 Table 3.3 shows the result of the same experiment performed on the Urban dataset. Here $\mathbf{p}_{\text{high}}$ =[0.9,0.75,0.7,0.65,0.8,0.85], $\mathbf{p}_{\text{mid}}$ = [0.8,0.7,0.65,0.6,0.75,0.8], and $\mathbf{p}_{\text{low}}$ = [0.7,0.6,0.55,0.51,0.65,0.7]. The noise levels are $\sigma = 0.001, 0.005$ and 0.01. The results show that

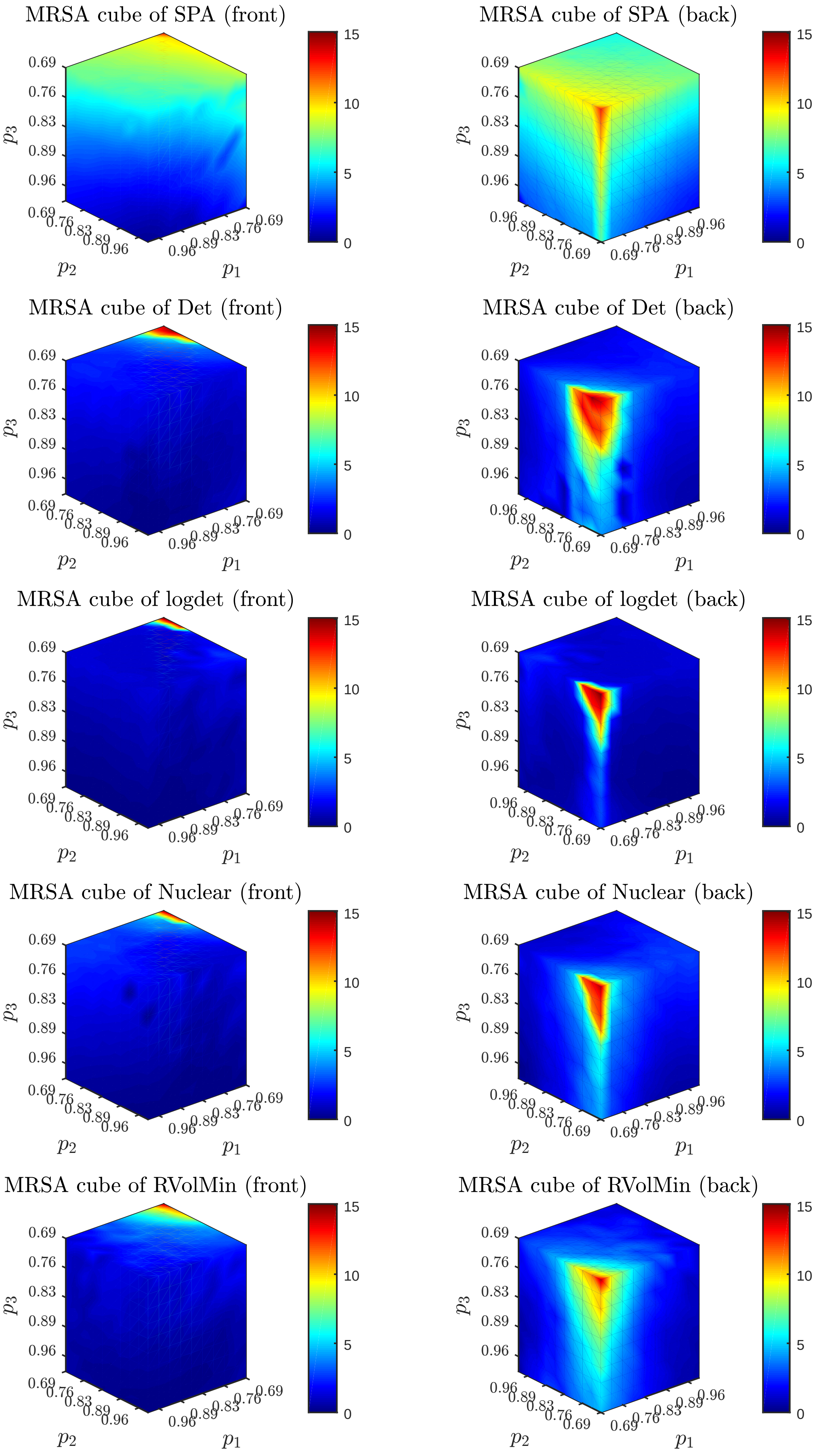


Fig. 3.7. MRSA cubes for different algorithms on synthetic Samson dataset, viewed from two angles.

- All the methods improve the fitting accuracy of SPA.
- Det has the best performance in most cases, with logdet as the first-runner up.
- Det performs well for $\mathbf{p}_{\text{high}}$ and $\mathbf{p}_{\text{mid}}$. With $\mathbf{p}_{\text{low}}$, it is not as good as logdet.
- Nuclear and RVolMin have the worst performances in all cases.
- The computational times: Det 7.5±0.2, logdet 1.8±0.1, Nuclear 1.5±0.0 and RVolMin 2.3±0.0, respectively.

On the Cuprite dataset with r = 12 Table 3.3 shows the result on the same experiments conducted on Cuprite dataset. Here we concatenate the $\mathbf{p}$ vector used in Urban two times to form the $\mathbf{p}$ vector with length 12 for the data Cuprite. e.g., for $\mathbf{p}_{\text{high}}^{\text{Cuprite}}$, we use $\mathbf{p}_{\text{high}}^{\text{Cuprite}} = [\mathbf{p}_{\text{high}}^{\text{Urban}} \ \mathbf{p}_{\text{high}}^{\text{Urban}}]$. The results from the tables show that

- All the methods improve the fitting accuracy of SPA.
- logdet has the best performance in all situations, with Det and Nuclear as the first-runner ups.
- RVolMin has the worst performances in most cases.
- The computational time: Det takes 28.1±0.9, logdet takes 3.8±1.1, Nuclear takes 3.4±0.1, and RVolMin takes 3.2±0.0. As expected, when r increase, Det takes much more time to run due to more QP's has to be computed, and therefore it is not recommended to be used for large r.

In general, the results show that the minvol NMF algorithms with Det and logdet perform well, better than RVolMin and Nuclear in terms of fitting accuracy. As RVolMin consistently produces inferior results, we exclude it in the subsequent sections.

Why RVolMin consistently performs poorer than minvol NMF? Here we give a short explanation why RVolMin consistently produces inferior results in almost all the experiments, even if the model formulation of RVolMin is similar to minvol NMF with $\mathcal{V}_{\text{logdet}}$. The key is about the choice of parameter δ: it leads to bad model and bad conditioning of the matrix that slow down the convergence. As discussed in §2.1.3 and §2.2.1 , the parameter δ should not be chosen too small: 1) a small δ greatly affects the convergence on solving the subproblem on $\mathbf{H}$ as the matrix $\mathbf{W}^\top\mathbf{W} + \delta\mathbf{I}$ now can be badly conditioned, and 2) a small δ gives too much importance to zero singular values of $\mathbf{W}$ which might not be desirable, c.f. Fig.2.1 . And in fact δ was chosen too small in RVolMin: 10^{-8} in [41] which, as explained in §2.1.3 and §2.2.1 , is not a desirable choice. The extensive experimental results in this section illustrated one important fact that, parameter tuning is one of the most critical aspect when applying machine learning algorithm in practice, and parameters should not be chosen arbitrarily.

3.3.3 On image segmentation on real HSI data and adding sparsity constraints on $\mathbf{H}$

Now we perform decomposition on the real HU data Samson and Jasper, to showcase the ability of minvol NMF to provide meaningful decomposition on HSI data. Here we give improved result over [6] on the algorithms Det, logdet and Nuclear on real HU data Samson and Jasper. We refer the reader to the last experiment in [6] for preliminary results.

Table 3.3: MRSA values for the three datasets.

On Jasper dataset with $r = 4$

	Across different $\mathbf{p}$ ($\sigma = 0.001$)			Across different noise levels ($\mathbf{p} = \mathbf{p}_{\text{high}}$)		
Method	$\mathbf{p}_{\text{high}}$	$\mathbf{p}_{\text{mid}}$	$\mathbf{p}_{\text{low}}$	$\sigma = 0.001$	$\sigma = 0.01$	$\sigma = 0.05$
`SPA`	5.40±0.60	12.62±0.18	20.76±0.23	5.40±0.60	7.29±0.51	24.59±1.43
`Det`	**0.41±0.08**	**0.40±0.06**	**10.99±1.68**	**0.41±0.08**	**0.74±0.06**	**4.90±3.27**
`logdet`	0.48±0.54	3.03±0.28	12.57±1.49	0.48±0.54	1.40±0.08	9.00±3.88
`Nuclear`	0.64±0.07	2.12±0.13	19.90±1.40	0.64±0.07	1.23±0.06	6.78±5.78
`RVolMin`	1.04±0.38	4.99±0.22	13.67±2.99	1.04±0.38	2.31±0.28	10.23±2.05

On Urban dataset with $r = 6$

	Across different $\mathbf{p}$ ($\sigma = 0.001$)			Across different noise levels ($\mathbf{p} = \mathbf{p}_{\text{high}}$)		
Method	$\mathbf{p}_{\text{high}}$	$\mathbf{p}_{\text{mid}}$	$\mathbf{p}_{\text{low}}$	$\sigma = 0.001$	$\sigma = 0.005$	$\sigma = 0.01$
`SPA`	7.83±0.93	10.32±1.53	16.22±2.00	7.83±0.93	8.56±0.86	15.95±3.57
`Det`	**0.54±0.11**	**2.45±1.25**	10.08±5.71	**0.54±0.11**	**1.25±0.48**	**6.85±3.47**
`logdet`	1.27±0.68	3.09±2.04	**8.78±2.58**	1.27±0.68	4.41±0.93	9.85±3.36
`Nuclear`	3.79±0.62	6.39±1.54	13.48±4.53	3.79±0.62	4.58±0.95	11.75±3.35
`RVolMin`	3.03±0.90	5.64±1.29	13.05±4.28	3.03±0.9	4.95±1.01	15.32±9.35

On Cuprite dataset with $r = 12$

	Across different $\mathbf{p}$ ($\sigma = 0.001$)			Across different noise levels ($\mathbf{p} = \mathbf{p}_{\text{high}}$)		
Method	$\mathbf{p}_{\text{high}}$	$\mathbf{p}_{\text{mid}}$	$\mathbf{p}_{\text{low}}$	$\sigma = 0.001$	$\sigma = 0.005$	$\sigma = 0.01$
`SPA`	6.59±0.98	8.87±1.42	11.53±1.39	6.59±0.98	7.24±1.07	11.53±1.39
`Det`	2.59±0.74	4.40±1.51	9.71±2.43	2.59±0.74	4.34±1.74	9.71±2.43
`logdet`	**2.51±0.59**	**3.85±0.96**	**8.41±2.04**	**2.51±0.59**	**4.01±1.22**	**8.41±2.04**
`Nuclear`	2.55±0.70	4.33±1.56	10.07±2.73	2.55±0.70	4.38±1.72	10.07±2.73
`RVolMin`	3.72±2.41	5.39±2.29	10.82±3.31	3.72±2.41	4.66±1.73	10.82±3.31

Adding sparsity constraint on columns of $\mathbf{H}$ As pointed out in [6], adding sparsity condition on the decomposition may help to obtain a cleaner abundance map. For example, it is expected that the pixels of watery region (such as river, pool or sea) contain just water. Therefore, the part of $\mathbf{H}$ corresponding to these components should be sparse.

Taking sparsity on $\mathbf{H}$ into account in the decomposition, we modify the minvol NMF model (2.1a) as follows. We add the L_1 norm of columns of $\mathbf{H}$ to the cost function of the minvol NMF (2.1a). This regularization promotes sparsity to all the columns of $\mathbf{H}$. Now focus on the jth column of $\mathbf{H}$, i.e., $\mathbf{h}_j$, for simplicity we hide the subscript j, this optimization subproblem can be expressed as

$$\underset{\mathbf{h}}{\operatorname{argmin}}\ \phi(\mathbf{h}) = \frac{1}{2}\|\mathbf{W}\mathbf{h} - \mathbf{m}\|_2^2 + \lambda\|\mathbf{h}\|_1 \quad \text{s.t.} \quad \mathbf{h} \geq 0,\ \mathbf{h}^\top \mathbf{1}_r \leq 1, \tag{3.6}$$

where $\lambda \geq 0$ is the regularization constant. Compared with the subproblem (2.16), here the cost function $\phi(\mathbf{h}_j)$ contains an extra regularizer term.

How to solve the new problem: iterative soft thresholding We now discuss the idea how to solve (3.6) . Ignoring the nonnegative constraint for a moment, the partial Lagrangian associated with the normalization constraint is

$$L(\mathbf{h}, \nu) = \frac{1}{2}\|\mathbf{W}\mathbf{h} - \mathbf{m}\|_2^2 + \lambda\|\mathbf{h}\|_1 + \nu(\mathbf{h}^\top \mathbf{1}_r - 1),$$

where $\nu \in \mathbb{R}$ is the Lagrangian multiplier. The solution to Problem (3.6) is given by solving the saddle-point problem

$$\min_{\mathbf{h}\geq 0} \max_{\mu} L(\mathbf{h}, \nu).$$

We solve such problem by alternating optimization: solve the subproblem on $\mathbf{h}$ while fixing ν, and then solve the subproblem on ν while fixing $\mathbf{h}$, and repeat the whole process until converge. The subproblem on μ is easy to solve, so we focus on solving the subproblem on $\mathbf{h}$:

$$\operatorname*{argmin}_{\mathbf{h}\geq 0} L(\mathbf{h}) = \frac{1}{2}\|\mathbf{W}\mathbf{h} - \mathbf{m}\|_2^2 + \lambda\|\mathbf{h}\|_1 + \nu(\mathbf{h}^\top \mathbf{1}_r - 1). \tag{3.7}$$

Problem (3.7) can be solved by accelerated proximal gradient. The idea is to first apply gradient update on the smooth part of $L(\mathbf{h})$ to obtain a temporary variable $\mathbf{h}^*$, then apply the proximal operator on the point $\mathbf{h}^*$ with respect to the nonsmooth part $\lambda\|\mathbf{h}\|_1$. It can be shown that the proximal gradient update of such problem has closed-form expression [10], which is a soft-thresholding step.

Hard constraint on columns of H and hard thresholding Inspired by iterative soft thresholding update for solving subproblem (3.6), we also consider an heuristic that use hard thresholding on the column of $\mathbf{H}$. That is, each time after the subproblem (3.7) is solved, we perform a N-sparse hard thresholding step: we keep the largest N entries in the vector untouched, and set all the rest to zero.

Result We now discuss the result the minvol NMF with and without the sparsity on $\mathbf{H}$ on the HU application. Fig. 3.8 and Fig. 3.9 show the decompositions. The reference result is provided by [115], in which the data are preprocessed, and the minimization model employed a sparsity regularization on $\mathbf{H}$ to make the abundance map cleaner. As the goal here is to showcase the ability of minvol NMF to provide meaningful decomposition, so, as stated in section 3.3.1 , when we solve minvol NMF problems on the data, we use the raw data directly as it is, no pre-processing is performed. Furthermore, we do not perform detailed parameter tuning in this example. The result illustrated the ability of minvol NMF to provide a meaningful decomposition, and adding additional modeling constraint helps to get a better result.

3.3.4 On handling spectral variability

As mentioned in §3.1 , endmembers are defined as macroscopically pure components in the image, while in real-world data, such assumption may fail due to Spectral Variability, which is, “the shallow water and deep water are different”. In this subsection we give an empirical evidence that minvol NMF can handle such issue by simply using a larger factorization rank r in the decomposition. We use the Samson dataset for illustration. Refer to Fig.3.4 , the dataset has three components: water, vegetation and dirt, hence r should be 3. However we decompose the data using minvol NMF (with 2-sparse constraint on columns of $\mathbf{H}$) with the factorization rank $r = 6$. The decomposition result contains two type of trees, and two type of water, see 3.10 . We can see that the “Sea shore” component is extracted, which consist of the mixture of water spectra and tree spectra. Lastly, we recombine components “Deep water” and “Shallow water” into “Water”, and “Tree type 1” with

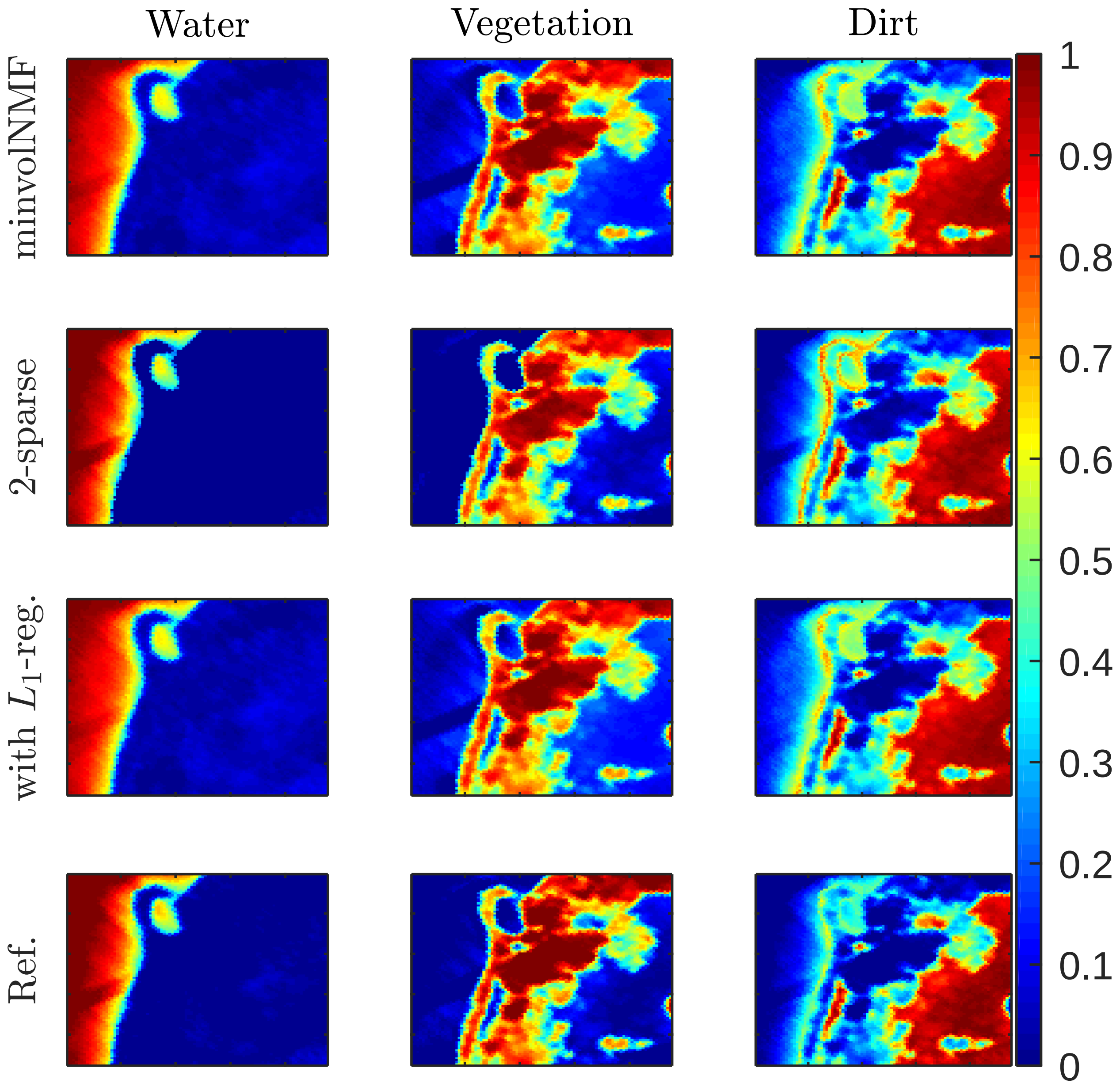


Fig. 3.8. The abundance map of the decomposition of the dataset Samson. The 2-sparse model has a cleaner abundance map for the component "water'. Furthermore, we can see that the sea shore components in all minvol NMF models are clearer than the reference one.

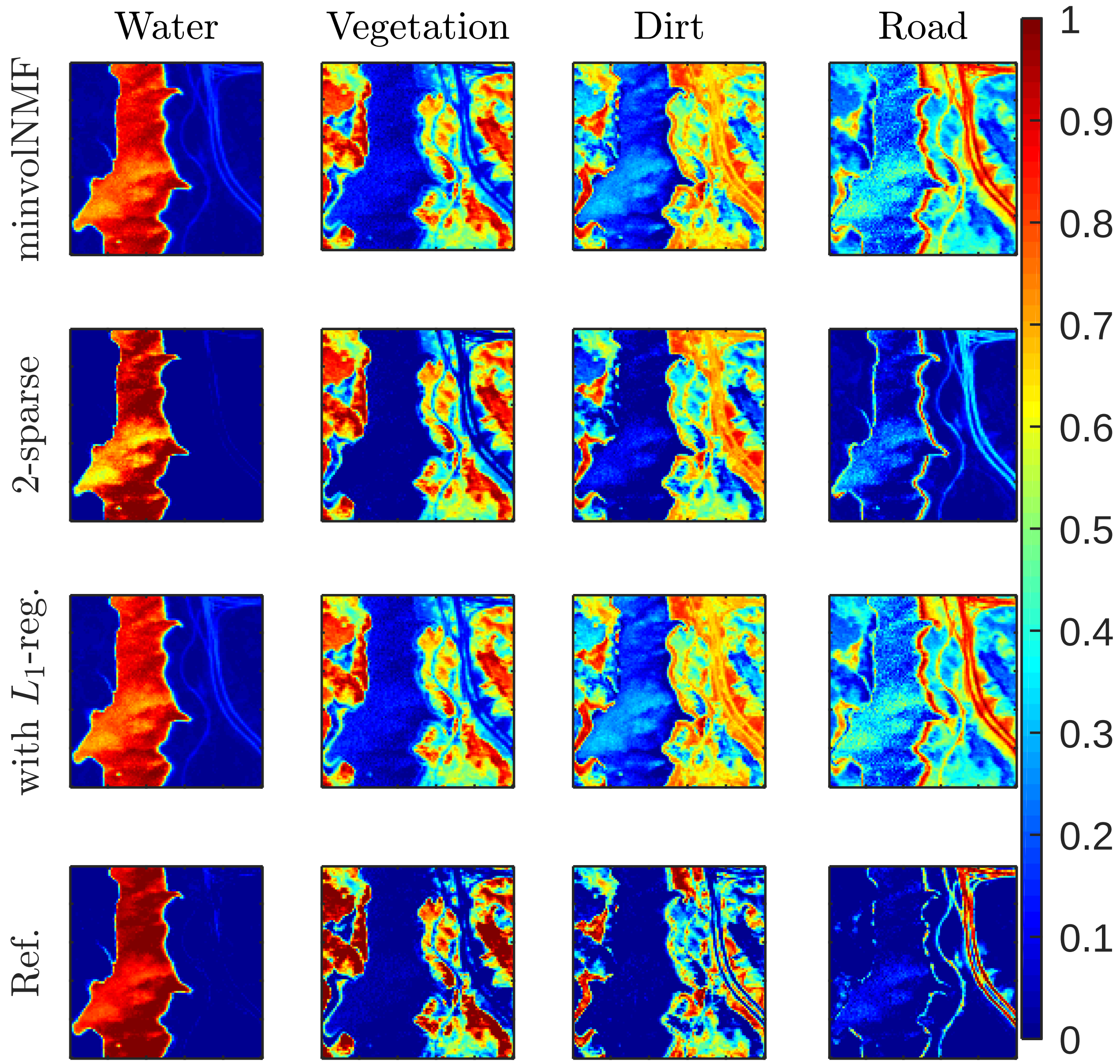


Fig. 3.9. The abundance map of the decomposition of the dataset Jasper. The 2-sparse model has a cleaner abundance map for the component “water’. Furthermore, we can see that the sea shore components in all minvol NMF models are clearer than the reference one.

“Tree type 2” into “Tree”. The resulting abundance maps have a clearer boundary than those in the reference.

3.4 Chapter summary and perspectives

Chapter summary We introduced Hyperspectral Unmixing (HU) as one of the application of NMF. The experimental results showed that minvol NMF algorithms are superior to other algorithms in HU problems on both synthetic and real datasets. In particular, the minvol NMF algorithms outperform both the state-of-the-art separable NMF algorithm `SPA`, and two volume-based methods such as `MVC-NMF` [84] and `RVolMin` [41]. The minvol NMF with determinant volume and log-determinant (logdet) volume are the top two methods among all the tested methods, where the logdet-based one is better than the det-based one.

Open problems We now state the opening problems concerning the minvol NMF on HU.

On handling spectral variability In §3.3.4 , we illustrated the ability of minvol NMF on handling spectral variability, the approach is to use over-estimated r in the factorization, followed by recombination of the components. The result so far is only preliminary, two questions remain open: How to choose an appropriate overestimated rank r? How to recombine the components? Answering these questions will greatly enhance the effectiveness of applying minvol NMF in HU applications.

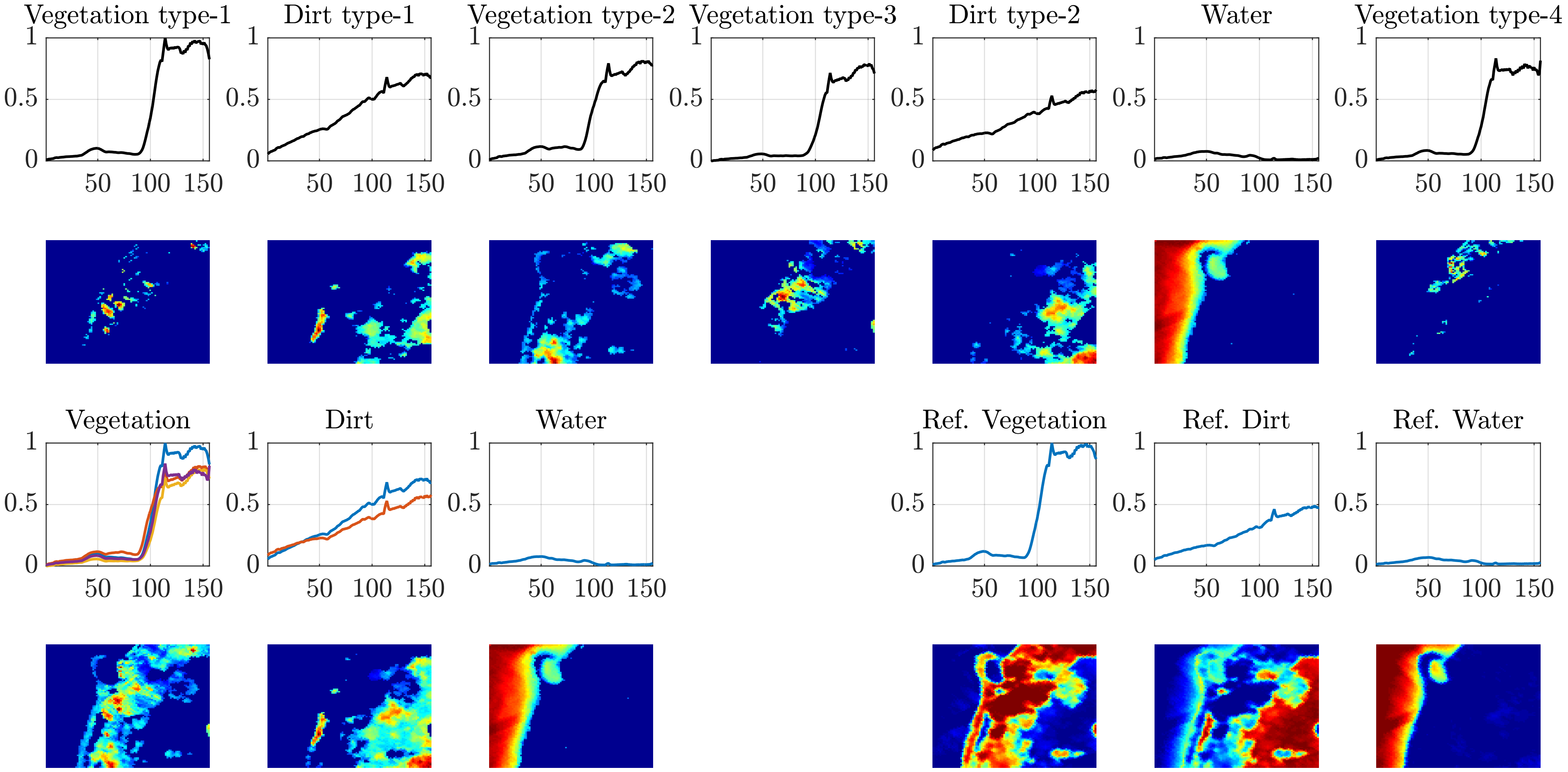


Fig. 3.10. **Top**: The abundance map of the decomposition of the Samson dataset with $r = 7$. **Bottom left**: The recombination of the components into $r = 3$. **Bottom right**: Reference.

4 Minvol NMF on audio source separation

All models are wrong; but some are useful.

George E. P. Box

In this chapter, we discuss NMF on the application of audio *Blind Source Separation* (BSS). We treat NMF as a black box and we do not focus too deep on the details on the algorithm design here, rather, we focus on how NMF can be used to solve audio BSS problems. For the details on the NMF algorithms design, see [72, 73].

Chapter organization We first give a brief introduction about audio BSS in §4.1 , then we look at how NMF can be used to solve audio BSS problem in §4.2 , where we provide a few examples on decomposing piano recording.

Highlights of contributions The main purpose of this section is to showcase the effectiveness of NMF-based models on BSS problem, which is another application domain where NMF shines. After giving the preliminary material on introducing audio BSS problem in §4.1 , we apply minvol NMF model (2.1b) for the audio BSS problem. To be more suitable for audio data, we changed the data fitting term from Frobenius norm to the β-divergence, where we focus on the case $\beta = 1$, where we name the model minvol KL-NMF. We then present numerical result in §4.2 , illustrating the effectiveness of minvol KL-NMF on audio BSS problem. Notice that the papers [72, 73] are collaborative work of the author and his coauthors, we do not present every material from [72, 73].

4.1 Single channel audio BSS

The task of audio BSS is similar to the goal of HU discussed in §3.1 . In audio BSS, the goal is to extract unknown signals called sources from the data, which is in the format of audio recording. Here, we consider a single channel recording, i.e., the audio signal is captured by a single microphone. The data here is a vector $\mathbf{x} \in \mathbb{R}^T$, where T denotes the duration of time points.

As audio BSS problems are similar to HU problems, we use Table 4.1 , which is similar to Table 3.1 , to list the typical goals in audio BSS. We do not repeat the problem description on Table 4.1 for audio BSS. Furthermore, as stated in §1.4.3 , we assume r is given, so our focus is mainly tasks (2) and (3) in Table 4.1 .

Table 4.1: Three typical goals in audio BSS.

	Audio BSS objective	Corresponding task in NMF
(1)	Identify the number of sources.	Find r.
(2)	Obtain the spectral profile of the sources.	Find $\mathbf{W}$.
(3)	Identify the amount of activation of each sources.	Find $\mathbf{H}$.

To understand how and why NMF can be used to solve audio BSS problems, we first discuss what is the flow of a BSS process; see Fig.4.1 . We will explain each step in the figure in the following paragraphs.

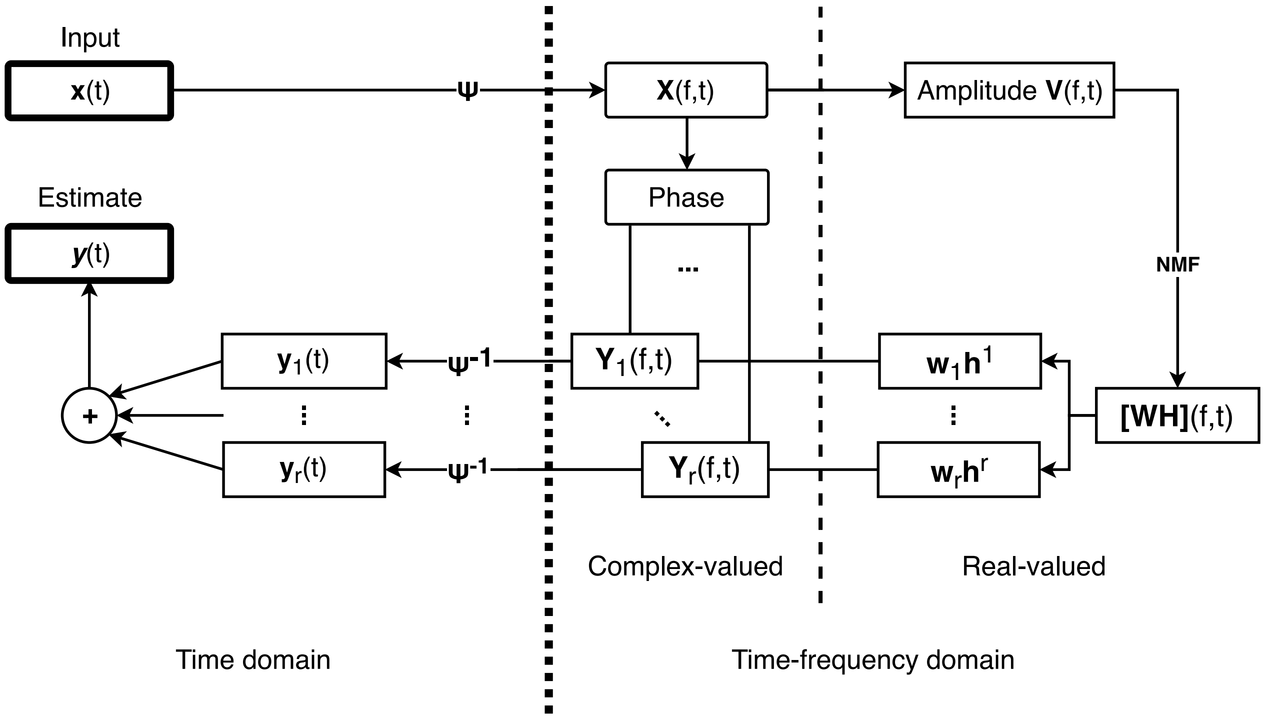


Fig. 4.1. The flow diagram of BSS of audio data using NMF.

4.1.1 The spectrogram

Given an audio recording $\mathbf{x} \in \mathbb{R}^T$ in time domain, a spectrogram that represents $\mathbf{x}$ in the time-frequency domains, is obtained by performing a *Time-Frequency Transform* (TFT) on $\mathbf{x}$. Examples of TFT are *(Discrete) Short Time Fourier Transform* (STFT) and *Discrete Wavelet Transform* (DWT) [83]. We denote a TFT operator as Ψ, and in this thesis, for simplicity, all TFT operators are STFT. For more discussion of STFT, see for example [70, 104] .

The TFT operator Ψ converts the vector $\mathbf{x} \in \mathbb{R}^T$ into a matrix $\mathbf{X} \in \mathbb{C}^{F \times T'}$ where F is the number of frequency bins and T' is the number of time points in $\mathbf{X}$. The value T' is affected by the use of smoothing window in the TFT operator. For example, for a window that uses 50% of the data point in $\mathbf{x}$, i.e., a window with 50% overlapping, $F \times T' \approxeq 2T$.

Properties of the spectrogram: sparse and low-rank The time-frequency representation $\mathbf{X}$ of a signal $\mathbf{x}$ is a matrix that is low-rank [104]. Such matrix has two fundamental properties that makes it low-rank: sparsity and redundancy. Redundancy comes from the fact that the frequency patterns of the signal repeats itself over time, while sparsity comes from two facts: many real-world audio signals are "banded", i.e., the energy of the signal only concentrates within a small frequency band; and many frequency bands in the whole spectrum are not active most of the time. For example, in a piano music, in each time frame, many piano keys are not pressed, i.e., the notes these unpressed keys represent are not active in that selected time frame. And for a single piano note, it consists of the fundamental harmonics and overtones, where most of the energy of the note is concentrated at the fundamental harmonics and the low overtones.

Mathematically, these two facts imply that the spectrogram of an audio is well approximated with a low-rank sparse matrix, in which NMF can be used to extract meaningful component in the audio data.

4.1.2 A simple linear mixing model and assumptions

Suppose the recording $\mathbf{x}$ is a mixture of r sources signals $\mathbf{s}_i \in \mathbb{R}^T$, $i \in [r]$, a goal in BSS is to estimate $\mathbf{s}_i$ from $\mathbf{x}$. Let the estimator of $\mathbf{s}_i$ be $\mathbf{y}_i$, then the BSS is perfect if $\|\mathbf{s}_i - \mathbf{y}_i\|_2 = 0$ for all i.

To estimate $\mathbf{s}_i$, we assume both the acquisition process and the mixing process are well modeled by a linear instantaneous mixing model:

$$\mathbf{x} = \sum_{i=1}^{r} \mathbf{s}_i. \tag{4.1}$$

That is, we assume:

- **Instantaneous mixing**. When time delay is present, the mixing can be modeled as $\mathbf{x}(t) = \sum_i \mathbf{s}_i(t - \tau_i)$ where τ_i is the delay time. Here we assume all $\tau_i = 0$.
- **No reflection and reverberation**, and thereby no non-linearity effect in the model. Similar to §3.1 where we discussed the hyperspectral unmixing problem, it is possible that there is non-linearity in the mixing process, but that is out of the scope of this thesis.

Note that we also implicitly assumed **balanced source**: the audio volume of the sources are equally the same, no source is particularly louder than others. If a source is too loud, it may become difficult to extract the weaker sources, and the recording can become clipped due to microphone saturation.

4.1.3 Time-frequency transform and decomposition

Now, let Ψ be a TFT operator (such as STFT). Applying Ψ on $\mathbf{x}$ gives the spectrogram

$$\mathbf{X} = \Psi(\mathbf{x}) \overset{(4.1)}{=} \sum_{i=1}^{r} \Psi(\mathbf{s}_i) = \sum_{i=1}^{r} \mathbf{S}_i, \text{ where } \mathbf{S}_i := \Psi(\mathbf{s}_i),$$

which is a matrix of complex numbers. Recall that a complex number $c = a + ib$ can be expressed in the polar form as $c = r\exp(i\theta)$. The matrix $\mathbf{X}$ can also be expressed in polar form as:

$$\mathbf{X}(f,t) = \mathbf{V}(f,t)\exp\big(i\Phi(f,t)\big) \in \mathbb{C}^{F\times T'}, \quad \text{for all } f \in [0,F] \text{ and } t \in [0,T]. \tag{4.2}$$

From $\mathbf{X}$ we define the amplitude spectrogram $\mathbf{V} \in \mathbb{R}_+^{F\times T'}$ as

$$\mathbf{V} = |\mathbf{X}|, \text{ where } \mathbf{V}(f,t) = |\mathbf{X}(f,t)| \text{ for all } f \in [0,F] \text{ and } t \in [0,T].$$

The part $\exp\big(i\Phi\big)$ in $\mathbf{X}$ is called the phase. Fig.4.2 shows an example of spectrogram.

To identify the sources, we decompose $\mathbf{X}$ using NMF, then we apply the inverse TFT on it to get $\mathbf{y}$. For a good BSS, $\mathbf{y}$ should be very close to $\mathbf{s}$. Precisely, as shown in Fig.4.1, we do not apply NMF directly on $\mathbf{X}$, but we apply NMF on $\mathbf{V}$.

When performing NMF on $\mathbf{V}$, in fact we assume

- **Additive mixture**. There is no sound cancellation between the sources. Mathematically, $\mathbf{V} = \sum_{i=1}^{r} |\mathbf{S}_i|$. This is true in many signals, and
- **Source is rank-1**. The source spectrograms $|\mathbf{S}_i|$ are well approximated by nonnegative rank-1 matrices, and hence a rank-r NMF corresponds to unmixing r sources. Note that this is a strong assumption and it makes sense only for simple audio such as a pure musical note shown in Fig.4.2. This assumption may fail for complicated signal such as human speech, in which a

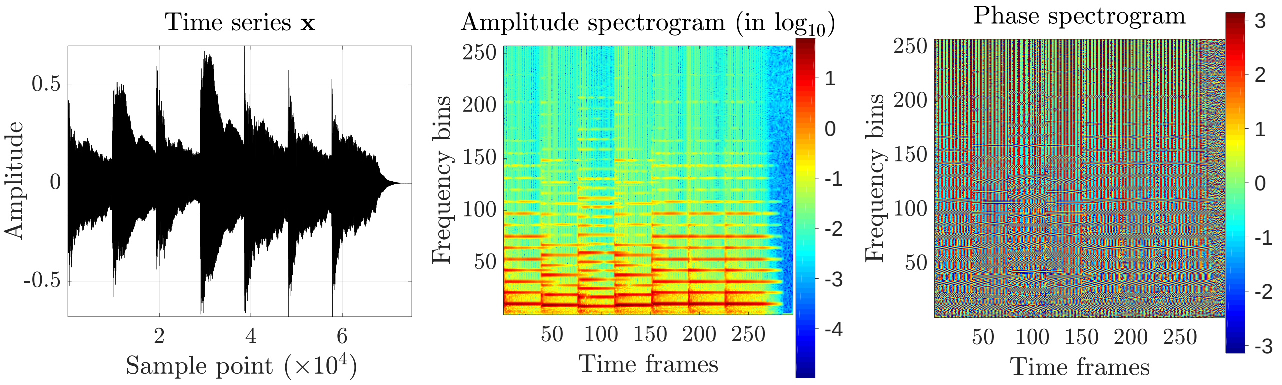


Fig. 4.2. The time series and the spectrograms of the audio recording of "Mary had a little lamb". See §4.2 for description of the data.

source can be made of several rank-1 factors. In this case we need a post-processing step to recombine them a posteriori (e.g., looking at the correspondence in the activation of the sources over time), or we can consider the grouped NMF. We discuss this issue further in §4.2.2 when we perform an experiment on a tabular bell audio.

With this setup, we present the NMF model in the next subsection.

Remark 2. *We give a few remarks on the above modeling.*

- *Let $\Psi^\dagger$ be the conjugate transpose of Ψ, then $\Psi^\dagger\Psi = F\mathbf{I}$ [70, Section 1.2.2], meaning that $\Psi^\dagger$ is the inverse operator of Ψ. Note that here the notion of "inverse" is not in the mathematical sense, since in general Ψ is not a surjective mapping from $\mathbb{R}^T$ to $\mathbb{C}^{F\times T'}$, i.e., it is possible that some spectrograms do not correspond to any TFT of a real signal. This issue is called consistency in time frequency analysis [65]. We discuss this further in §4.1.5 .*

- *Here we focus on the NMF stage of the BSS process on factorizing the amplitude spectrogram* $\mathbf{V}$ *into the source amplitude spectrograms. For the phases reconstruction, which is a highly non-trivial problem, we consider a naive reconstruction procedure consisting in keeping the same phase as the input mixture for each source. We come back to this issue in §4.1.6 .*

4.1.4 KL-NMF for audio source separation

Given a nonnegative spectrogram matrix $\mathbf{V} \in \mathbb{R}_+^{F\times T'}$ and the factorization rank r (the number of sources), we solve the audio BSS problem by solving the corresponding NMF problem, which produces $\mathbf{W}$ and $\mathbf{H}$ such that $\mathbf{V} \approxeq \mathbf{WH}$. For $\mathbf{V}$ representing the amplitude spectrogram of an audio signal, the matrix $\mathbf{W}$ is called the dictionary and each column corresponds to the spectral content of a source, and $\mathbf{H}$ is the activation matrix specifying if a source is active at a certain time frame and in which intensity. That is, each rank-one factor $\mathbf{W}(:,i)\mathbf{H}(i,:)$ corresponds to a source. To compute $\mathbf{W}$ and $\mathbf{H}$, we solve the following NMF

$$\min_{\mathbf{W},\mathbf{H}} D\left(\mathbf{V}|\mathbf{WH}\right) = \sum_{f,t} d\Big(\mathbf{V}_{ft}|[\mathbf{WH}]_{ft}\Big) \quad \text{s.t. } \mathbf{W}\geq 0, \mathbf{H}\geq 0, \tag{4.3}$$

where $d(x|y)$ is the discrete β-divergence:

$$d(x|y) = \begin{cases} \frac{1}{\beta(\beta-1)} \left(x^{\beta} + (\beta-1)\, y^{\beta} - \beta x y^{\beta-1} \right) & \text{if } \beta \neq 0, 1, \\ x \log \frac{x}{y} - x + y & \text{if } \beta = 1, \\ \frac{x}{y} - \log \frac{x}{y} - 1 & \text{if } \beta = 0. \end{cases} \tag{4.4}$$

About the β-divergence This function generalizes the Frobenius norm, which is a special case when $\beta = 2$. For $\beta = 1$ and $\beta = 0$, the β-divergence corresponds to the Kullback-Leibler (KL) divergence and the Itakura-Saito (IS) divergence, respectively. The β-divergence is homogeneous in β: i.e., $d(\lambda x|\lambda y) = \lambda^{\beta} d(x|y)$. It implies that factorizations obtained with $\beta > 0$ will rely more heavily on the largest data values and less precision is to be expected in the estimation of the low-power components. The IS divergence ($\beta = 0$) is scale-invariant, i.e., $d_{IS}(\lambda x|\lambda y) = d_{IS}(x|y)$ [36]. The IS divergence is the only one in the β-divergences family to possess this property. It implies that time-frequency areas of low power are as important in the divergence computation as the areas of high power. This property is interesting in audio BSS as low-power frequency bands can perceptually contribute as much as high-power frequency bands. Note that both KL and IS divergences are more adapted to audio BSS than Frobenius norm as it is built on logarithmic scale as human perception [36]. Moreover, the β-divergence is only convex with respect to $\mathbf{W}$ (or $\mathbf{H}$) if $\beta \geq 1$. Otherwise, the objective function is nonconvex. This implies that, for $\beta < 1$, even the problem of inferring $\mathbf{H}$ with $\mathbf{W}$ fixed is non-convex. For more details on β-divergences; see [21, Section 2.6] or [36] . For simplicity, we focus on the KL-divergence from now on.

Why KL-divergence but not the Frobenius norm Frobenius norm is less suitable for audio BSS. First we recall two facts:

1. the (squared-)Frobenius norm emphasizes more on the large component in a matrix, and

2. spectrogram consists of many harmonics (see see Fig.4.2 for an example), and it is when all the harmonics are activated together that they form the characteristic sound of the note.

Hence, the (squared-)Frobenius norm focuses more on the large components in the harmonics while ignoring other weak components, which is undesirable for audio data. Due to such reason, in audio signal processing, the data fidelity term is usually expressed using, KL-divergence (or IS-divergence), as it takes into account on the small components in the audio spectrogram due to the log expression; see Equation (4.4).

minvol KL-NMF for audio BSS To solve audio BSS: we consider minvol KL-NMF model (2.1b) where the data fitting term is replaced by the KL-divergence:

$$\text{minimize } D(\mathbf{V}|\mathbf{W}\mathbf{H}) + \lambda \text{logdet}(\mathbf{W}^{\top}\mathbf{W} + \delta \mathbf{I}_r) \ \text{ s.t. } \ \mathbf{W}^{\top}\mathbf{1}_m = \mathbf{1}_r, \mathbf{W} \geq 0, \mathbf{H} \geq 0.$$

In this model, $\mathbf{W}$ is normalized, meaning that the spectral profile of the source are of similar volume, which is a sensible assumption in application.

We use BCD to solve this optimization problem: we solve the subproblem on $\mathbf{W}$ while fixing $\mathbf{H}$, and then solve the subproblem on $\mathbf{H}$ while fixing $\mathbf{W}$. The key issue here is the minimization subproblem on $\mathbf{W}$, in which we use majorization minimization (MM) to solve it; see [72] for details of the derivation of the majorizer and the whole algorithm.

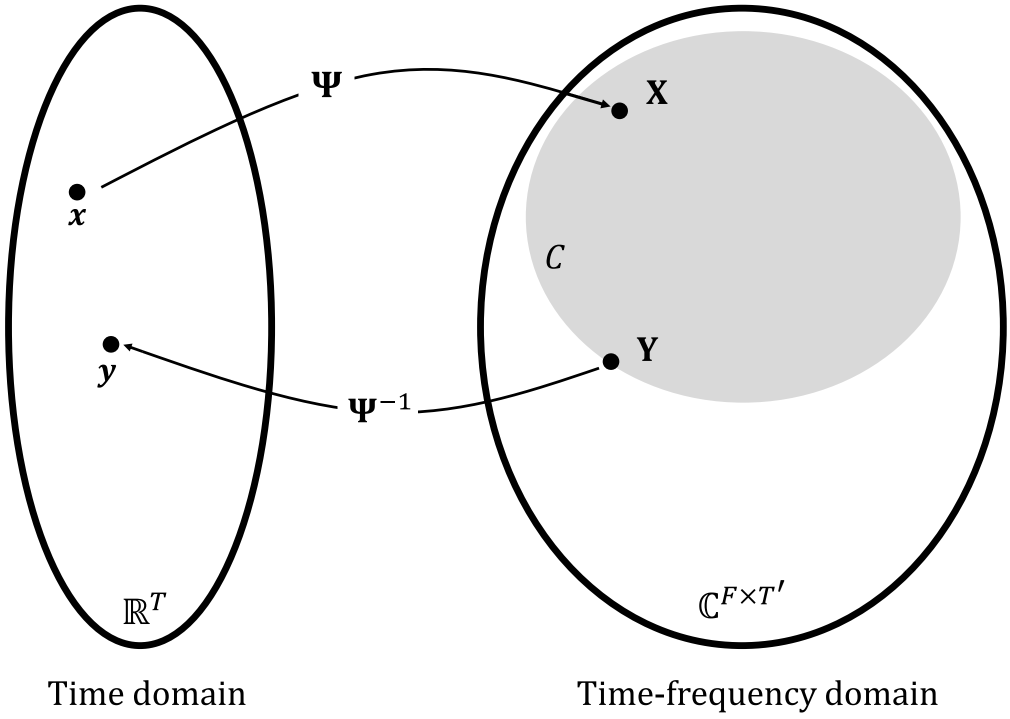


Fig. 4.3. A pictorial illustration of Lemma 4.1.2. A vector $\mathbf{x}$ in the time domain is mapped to a point $\mathbf{X}$ in C, and a point $\mathbf{Y}$ in C is mapped back to a point $\mathbf{y}$ in $\mathbb{R}^T$ by the inverse operator Ψ^{-1}. Note that the set C is only a proper subset of $\mathbb{C}^{F\times T'}$.

4.1.5 About consistency in time-frequency analysis

We now discuss an issue that can be very important in time-frequency analysis, called *Consistency.* To discuss such issue we first review two properties of the STFT operator Ψ.

Lemma 4.1.1. *(Ψ is injective) Given $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^T$, let $\mathbf{X}_1 = \Psi(\mathbf{x}_1)$ and $\mathbf{X}_2 = \Psi(\mathbf{x}_2)$. If $\mathbf{X}_1 = \mathbf{X}_2$, then $\mathbf{x}_1 = \mathbf{x}_2$.*

Proof. As Ψ^{-1} is linear, $\mathbf{x}_1 - \mathbf{x}_2 = \Psi^{-1}\mathbf{X}_1 - \Psi^{-1}\mathbf{X}_2 = \Psi^{-1}(\mathbf{X}_1 - \mathbf{X}_2) = \Psi^{-1}\mathbf{0} = \mathbf{0}$. □

Lemma 4.1.2. *(Image of Ψ is a proper subset of $\mathbb{C}^{F\times T'}$) Let the set of all time-frequency matrices produced by Ψ be C, defined as $C := \Big\{\mathbf{X} \in \mathbb{C}^{F\times T'} \mid \mathbf{X} = \Psi(\mathbf{x}), \mathbf{x} \in \mathbb{R}^T\Big\}$. Then $C \subsetneq \mathbb{C}^{F\times T'}$. In other words, Ψ is not surjective.*

Sketch of the proof By contradiction. Suppose $C \subseteq \mathbb{C}^{F\times T'}$ and $\Psi : \mathbb{R}^T \to \mathbb{C}^{F\times T'}$ is onto. By Lemma 4.1.1, Ψ is injective, then by open mapping theorem [38, Section 5.10], Ψ is an bounded bijective linear map and it has a bounded inverse. But Ψ^{-1} is not bounded, contradiction. □

Lemma 4.1.2 means that, the set of matrices that correspond to a vector in $\mathbb{R}^T$ is only a subset of $\mathbb{C}^{F\times T'}$, see Fig.4.3 .

Now, consider we perform NMF on a matrix $\mathbf{X}$, which is always in C. Using Equation (4.2), let $\mathbf{X} = \mathbf{V}\exp(i\mathbf{\Phi})$, then we perform NMF on $\mathbf{V}$. From here, there are two possibilities.

- (Case 1) Suppose NMF gives $\mathbf{V} = \mathbf{W}_1\mathbf{H}_1$ and let $\mathbf{Y} := \mathbf{W}_1\mathbf{H}_1\exp(i\mathbf{\Phi})$. Assume $\mathbf{Y} \in C$, then this decomposition is called *consistent* as $\mathbf{Y}$ corresponds to a $\mathbf{y} \in \mathbb{R}^T$.
- (Case 2) Suppose NMF gives $\mathbf{V} = \mathbf{W}_2\mathbf{H}_2$ and let $\mathbf{Z} := \mathbf{W}_2\mathbf{H}_2\exp(i\mathbf{\Phi})$. Assume $\mathbf{Z} \notin C$, then this decomposition is called *inconsistent*, as $\mathbf{Z}$ do not corresponds to any $\mathbf{y} \in \mathbb{R}^T$. From here, if

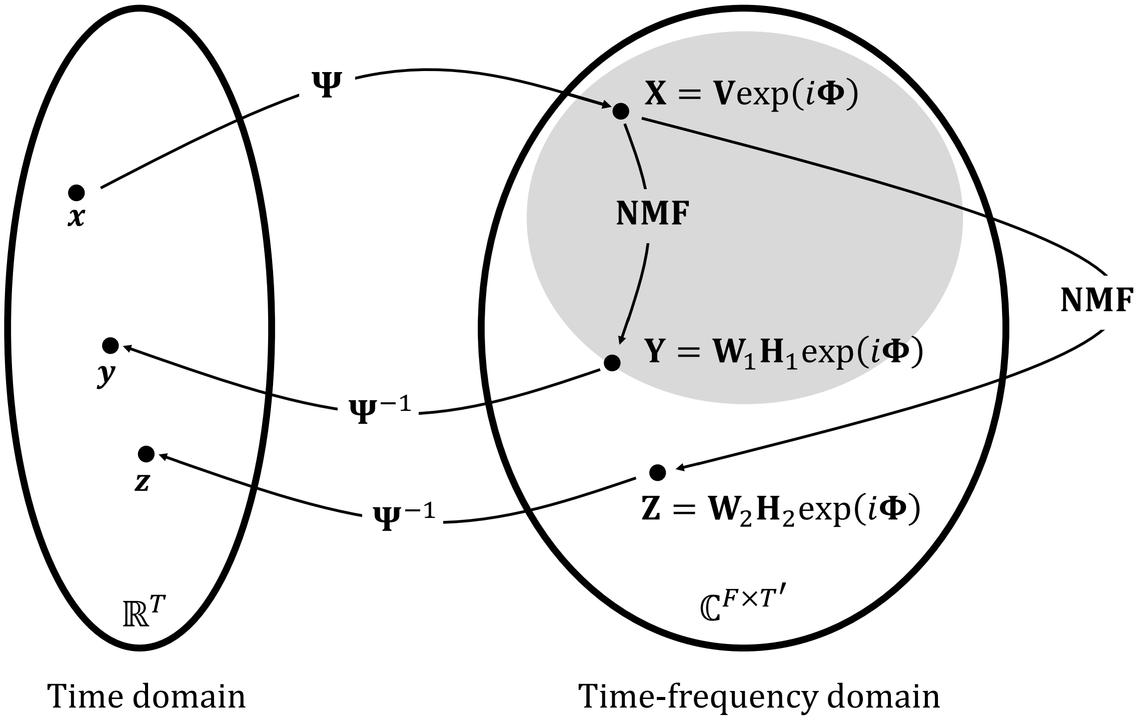

$\mathbf{\Psi}$
$\mathbf{X} = \mathbf{V}\exp(i\mathbf{\Phi})$
x
NMF
NMF
y
$\mathbf{Y} = \mathbf{W}_1\mathbf{H}_1\exp(i\mathbf{\Phi})$
$\mathbf{\Psi}^{-1}$
z
$\mathbf{\Psi}^{-1}$
$\mathbf{Z} = \mathbf{W}_2\mathbf{H}_2\exp(i\mathbf{\Phi})$
$\mathbb{R}^T$
$\mathbb{C}^{F\times T'}$
Time domain
Time-frequency domain

Fig. 4.4. Illustrating the consistency issue of decomposing a time-frequency matrix. The figure follows the same setup as Fig.4.3 . Here the NMF result $\mathbf{W}_1\mathbf{H}_1$ is consistent as the matrix $\mathbf{Y}$ while the result $\mathbf{W}_2\mathbf{H}_2$ is not.

we perform inverse TFT on $\mathbf{Z}$, we still get a vector $\mathbf{z} \in \mathbb{R}^T$, and in this case it is highly possible that $\mathbf{z}$ contains artifacts, where the artifacts are due to the non-surjectiveness of Ψ.

We illustrate the discussion above in Fig.4.4 .

Now we can say that, there is always an implicit assumption in applying NMF on time-frequency matrix, the assumption is that the spectrogram $\mathbf{WH}\exp\left(i\mathbf{\Phi}\right)$ has to be consistent. If this assumption is violated, then it is highly possible that the reconstructed vector in the time domain is contaminated by artifacts. Artifacts can seriously downgrade the quality of the solution, if the Signal-to-Artifact Ratio (SAR) is low. SAR is defined as follows. Let $\mathbf{v}, \mathbf{x}, \mathbf{a}$ denote the observed vector, the true signal and the artifact, respectively, and consider the model $\mathbf{v} = \mathbf{x} + \mathbf{a}$, then $SAR := \|\mathbf{x}\|_2^2/\|\mathbf{a}\|_2^2$.

I is very important to consider NMF under consistency constraint when SAR is low, and many research works on using NMF to decompose time-frequency matrices ignore this important issue, especially in those works on electroencephalogram analysis (see for examples [68, 69, 21]), where the power of the signal is well known to be very low.

In this thesis, we assume NMF provides consistent decompositions for the audio BSS problem, based on the fact that the data we study are relative clean and thus high SAR is expected. For NMF with consistency constraint, which is out of the scope of this thesis, we refer to the line of works by J. Le Roux [65, 64, 66].

4.1.6 About assumptions and violations

Table 4.2 summarizes all the assumptions we made so far in the BSS process. Violation of these assumptions may downgrade the effectiveness of applying NMF on audio BSS. We now briefly discuss about these assumption below.

Table 4.2: The summary of assumptions on using NMF for audio BSS.

A1	$\tau_i = 0$.
A2	Linear mixing model.
A3	Sources are balanced.
A4	$\mathbf{V} = \sum_i \lvert\mathbf{S}_i\rvert$.
A5	$\lvert\mathbf{S}_i\rvert$ are well approximated by nonnegative rank-1 matrices.
A6	Reconstructed components are consistent.
A7	The data has no outlier.

- A1 means the mixing is instantaneous. If there is time delay, a way to deal with it is to extend the NMF to Convolutive NMF [21, 28] which deal with time-correlated components.

- A2 means there is no nonlinear effect caused by reflection or reverberation. If it is violated, we can consider NMF model with non-linearity as discussed in §3.1.

- A3 means there is no clipping nor microphone saturation. If there is clipping or saturation, the unmixing task can be difficult, and we may need to try statistical approach which is out of the scope here.

- A4 means there is no sound cancellation between the sources. Violating this assumption basically rejects the use of NMF as NMF assumes the interaction between sources is additive. We can consider semi-NMF that only requires one factor (say $\mathbf{W}$) to be nonnegative.

- A5 refers to the cases that the audio is "sufficiently simple" such that the source spectrogram can be well approximated by nonnegative rank-1 matrices. This assumption can be violated when the audio is complicated. In this case we can consider the grouped NMF: instead of imposing each rank-1 component to be nonnegative, we impose the sub-patterns to be nonnegative. Mathematically, $\mathbf{V} = \mathbf{W}\mathbf{H} = \sum_i \mathbf{w}_i \mathbf{h}^i = \sum_k \mathbf{A}_k$ where $\mathbf{A}_k = \sum_i \mathbf{w}_i \mathbf{h}^i$ is nonnegative for some group indices $i \in g_k$. Notice that here it is $\mathbf{A}_k$ nonnegative but not $\mathbf{W}$ nor $\mathbf{H}$. To solve this problem, the technique from bounded matrix factorization [60] can be used.

- A6 is the consistency assumption discussed in §4.1.5 .

- If A7 is violated, we can consider robust NMF using robust norms.

All the above extensions of NMF are out of the scope of the thesis, but they are interesting directions to investigate.

4.2 Using minvol KL-NMF to solve single channel audio BSS problems

In this section we showcase the usefulness of minvol KL-NMF on audio BSS. For detail numerical comparisons on the general minvol β-NMF to other models, see [72].

Here we apply minvol KL-NMF on the spectrogram of two monophonic piano sequences. For the two monophonic piano sequences, the audio signals are true life signals with standard quality. Note that the sequences are made of pure piano notes, the number r should therefore correspond to the number of notes present into the mixed signals. The comparative study is performed for several values of r with a focus on the case where the factorization rank r is overestimated. For all simulations, random initializations are used for $\mathbf{W}$ and $\mathbf{H}$. In all cases, we use a Hamming window of size F=1024, and 50% overlap between two frames. The MATLAB code is available from `bit.ly/minvolKLNMF`.

4.2.1 Mary had a little lamb

The first audio sample is the beginning of the piece "Mary had a little lamb", see Fig.4.5 , and also Fig.4.2 for the spectrograms. The sequence is composed of 3 notes; E_4, D_4 and C_4. The recorded signal is 4.7 seconds long and down-sampled to f_s = 16000Hz yielding T=75200 samples. STFT of the input signal $\mathbf{x}$ yields a temporal resolution of 16ms and a frequency resolution of 31.25Hz, so that the amplitude spectrogram $\mathbf{V}$ has T'=294 frames and F=257 frequency bins. In the same figure, we also present the decomposition results on the amplitude spectrogram using minvol KL-NMF with different values of r. We observed the following.

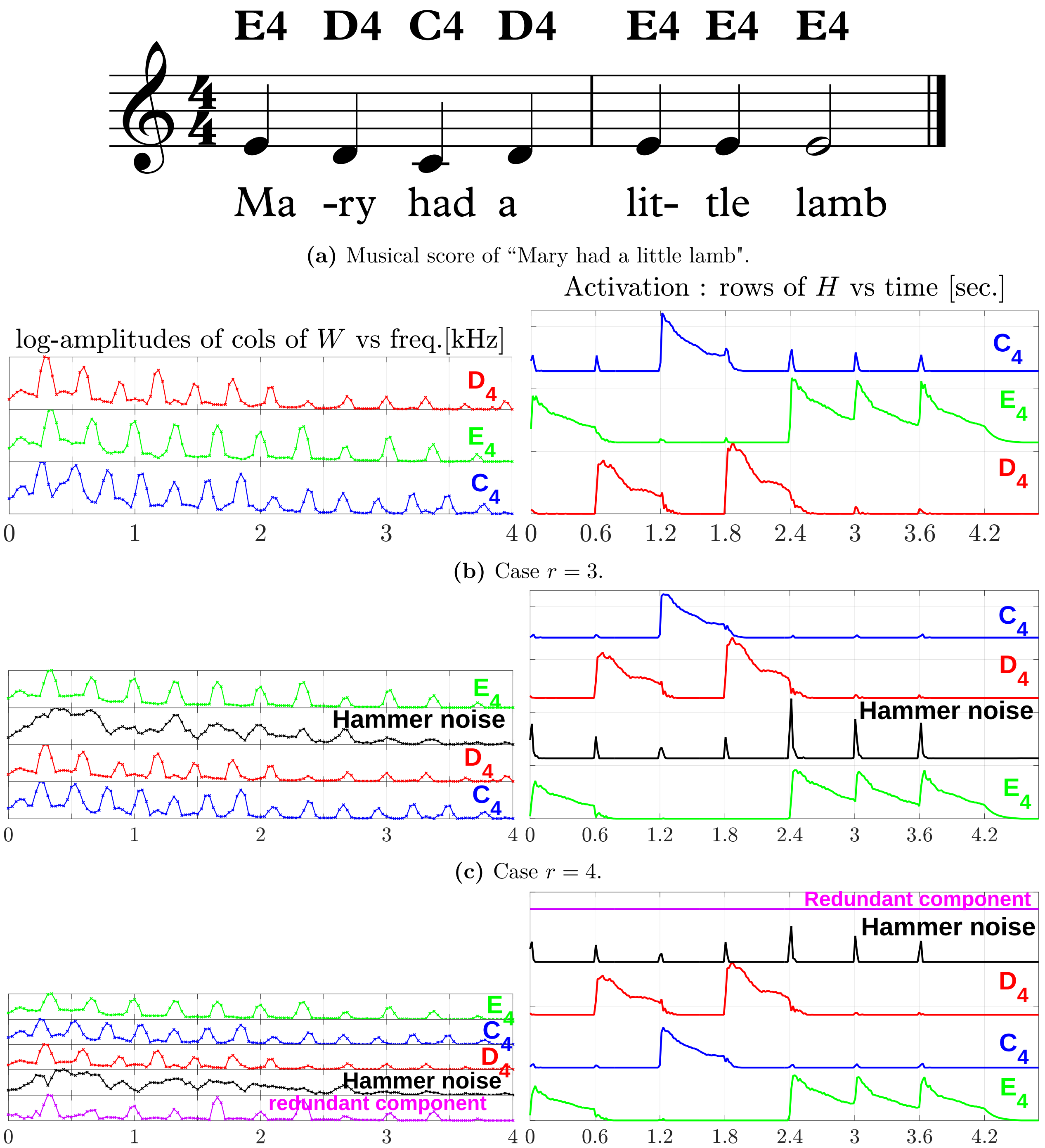
- In the case $r = 3$, all the simulations give a nice separation with similar results for $\mathbf{W}$ and $\mathbf{H}$. The activations are coherent with the sequences of the notes.

- In the case $r = 4$, we can see that minvol KL-NMF captured an component that corresponds to the noise within the piano just before triggering a specific note (in particular, the hammer noise). This observation is confirmed by the fact that the amplitude is proportional to the natural strength of the fingers playing the notes.

- In the case $r = 5$, it corresponds to the situation where the factorization rank is overestimated. We observe that minvol KL-NMF is able to extract the 3 notes correctly and set automatically to the redundant source estimates (more precisely, the corresponding row of $\mathbf{H}$ is set to zero, while the corresponding column of $\mathbf{W}$ have entries equal to one another as $||\mathbf{W}(:, i)||_1 = 1$). It is interesting to note that, here factorization rank is overestimated, while minvol KL-NMF can set the extra components to zero, and thus providing the ability to automatically select the model order. In fact, this phenomenon was first observed in the conference work [71], and then the work [72]. In [72], we tried the same experiments with $r = 7$, which is a factorization rank that is even more overestimated, and the same observation was made: the minvol KL-NMF can set the extra components to zero.

4.2.2 On a simple tubular bell music and the effectiveness of rank-1 approximation of each individual source

We now perform a similar experiment as the previous one on a Belgian hymn called "El Doudou", which is the music of the Doudou Festival of Mons, Belgium. Here the data is an synthesized audio of music played by tubular bells, and the audio is simulating the tabular bells inside a clock tower (called "belfry" in architecture). The note produced by the bell sustains much longer than that of piano, hence the spectrogram is more complicated as the notes are interrupted by previous notes. The music consists of 6 type of notes: $F_4, A_4, B\text{b}_4, C_5, D_5, E\text{b}_5$, and we uses minvol KL-NMF with overestimated rank $r = 8$ for the decomposition. We obtain a result similar to that of the previous experiment: with $r = 8$, minvol KL-NMF decomposes the audio into components that correspond to each of the music note, captures the hammer noise, and finds that one component is redundant. The experiment result is available at `https://www.youtube.com/watch?v=1BrpxvpghKQ`.



(a) Musical score of "Mary had a little lamb".

(b) Case $r = 3$.

(c) Case $r = 4$.

(d) Case $r = 5$.

Fig. 4.5. Minvol KL-NMF applied to "Mary had a little lamb" amplitude spectrogram with various r. When $r = 3$, the decomposition extracted three components in the audio that correspond exactly to the three notes in the score. When $r \geq 4$, an additional component that corresponds to the "hammer noise" is extracted. When $r = 5$, we see there is an component that is redundant, where the 5th row of $\mathbf{H}$ (the purple line) is zero, this case illustrated the ability of KL-minvol NMF to automatically setting redundant component to zero when r is overestimated.

Further inspection on the spectrogram of the audio shows that assumption A5 in Table 4.2 is violated. Refer to Fig.4.6 , each temporal block (between two hammer time) is in fact not rank-1. However, using a rank-1 nonnegative component here gives an estimate that is good enough. This observation motives the design of grouped NMF as discussed in §4.1.6 .

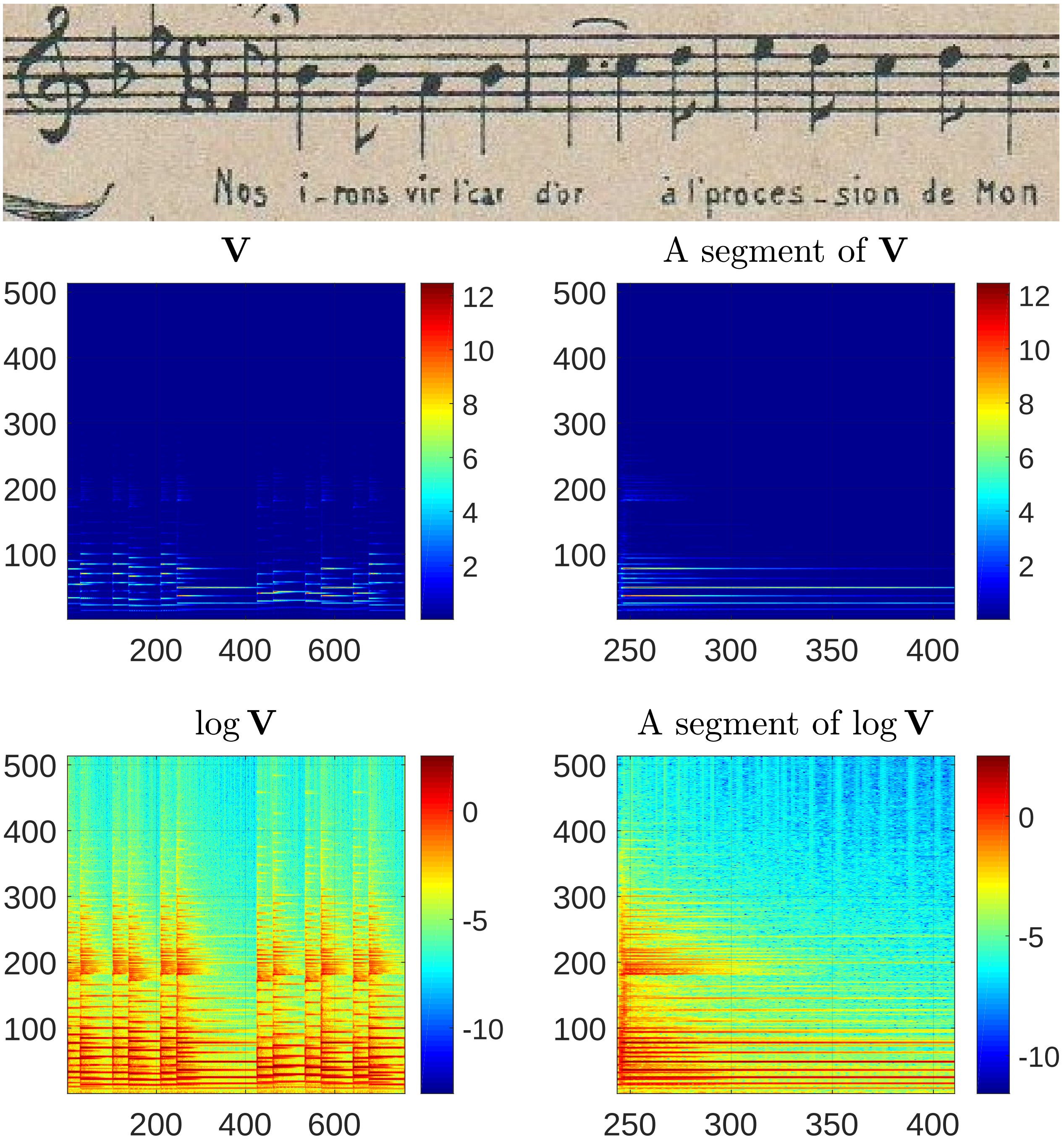


Fig. 4.6. The score and the amplitude spectrogram of the music "El Doudou". **Top**: the music score. **Bottom**: the amplitude spectrograms, plotted in linear scale and in log scale. Here we do not specify the exact values of the axes and just express them as index. In the second column, we show a temporal block of the amplitude spectrogram (correspond to the C_5 note in the score) with x-axis zoomed between indices 244 and 410. The log-plot show that the block is not exactly rank-1, hence assumption A5 in Table 4.2 is violated.

4.2.3 Simple chord

We now apply minvol KL-NMF on a simple chord audio, see Fig.4.7. Three notes are chosen: C_4, E_4 and G_4, they are played together in different time instances with different dynamics (loudness): **f** (forte, meaning "strong") and **p** (piano, meaning "soft"). As the notes are played together, the harmonics are completely overlapped with each other and hence the spectrogram are much more complicated compared with the previous two experiments. Minvol KL-NMF with $r = 3$ is able to extract the 3 notes correctly, and the rows of $\mathbf{H}$ corresponds to the dynamics in the music score.

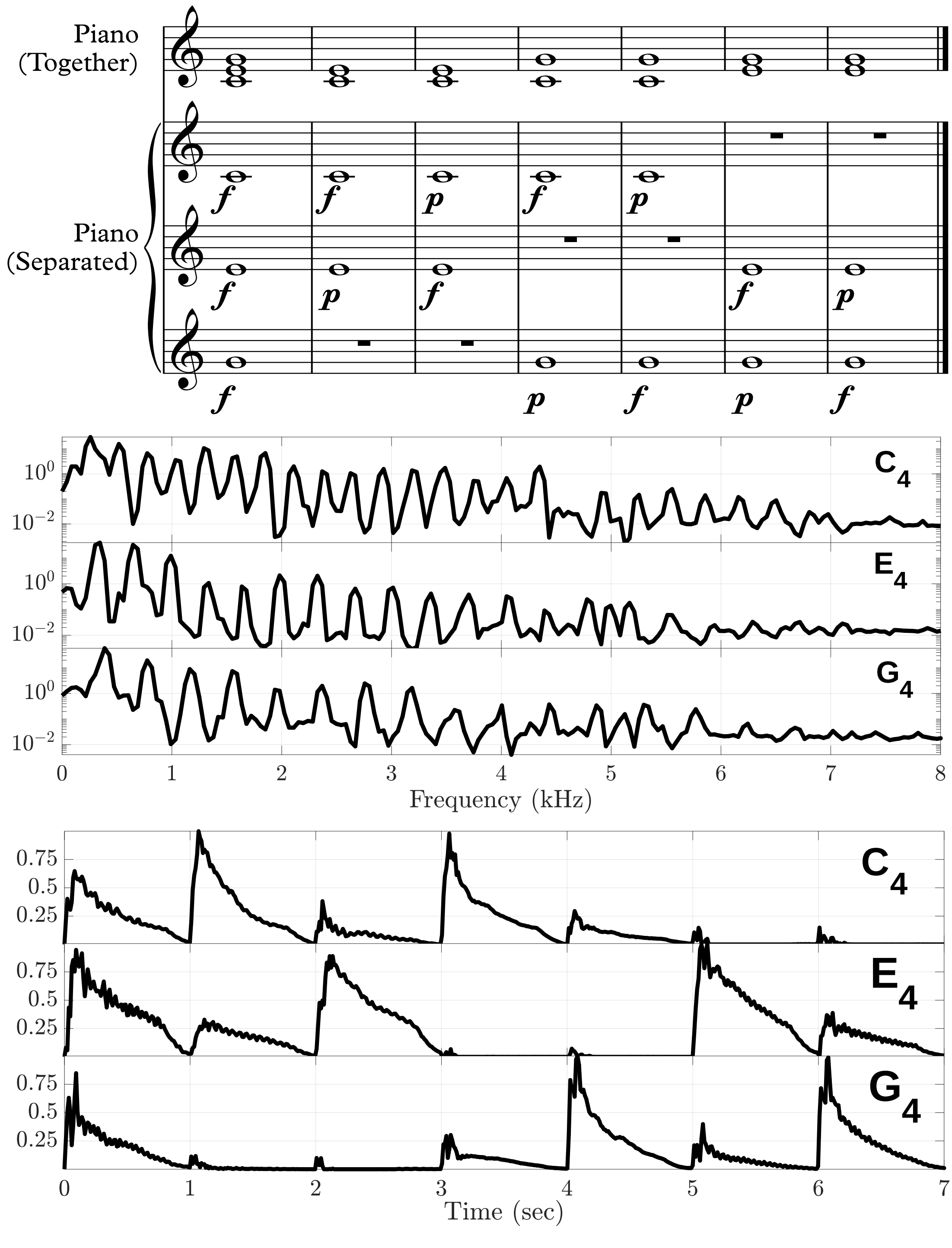


Fig. 4.7. Minvol KL-NMF applied to a chord amplitude spectrogram. **Top**: the music score (with notes written together and separately). **Bottom**: the columns of $\mathbf{W}$ and the rows of $\mathbf{H}$.

4.2.4 Prelude of Bach

We now apply minvol KL-NMF on a more complicated audio. We use the first 30 seconds of "Prelude and Fugue No.1 in C major" from J. S. Bach played by Glenn Gould[1], see Fig. 4.8 for the score. The audio sample is a sequence of 13 notes: B_3, C_4, D_4, E_4, $F_4^{\#}$, G_4, A_4, C_5, D_5, E_5, F_5, G_5, A_5. No preprocessing such as noise removal is applied on the sample. The recorded signal is down-sampled to $f_s = 11025$Hz yielding T=330750 samples. STFT of the input signal $\mathbf{x}$ yields a temporal resolution of 46ms and a frequency resolution of 10.76Hz, so that the amplitude spectrogram $\mathbf{V}$ has T'=647 frames and F=513 frequency bins.

Fig. 4.9 present the decomposition results on the amplitude spectrogram using minvol KL-NMF with $r = 16$, which is overestimated. We observe that minvol KL-NMF successfully detected the 13 source estimates corresponding to each of the note, obtained one component (the 8th) which corresponds to the hammer noise, and automatically sets 2 components to zero (6th and 13th). The analysis of the fundamentals (maximum peak frequency) of the 13 source estimates correspond to the theoretical fundamentals of the 13 notes mentioned earlier. Additionally, the activations are coherent with the sequences of the notes. Figure 4.10 shows (on a limited time interval) that the estimate sequence follows the sequence in the score. Note that a threshold and permutations on rows of $\mathbf{H}$ is used to improve visibility.

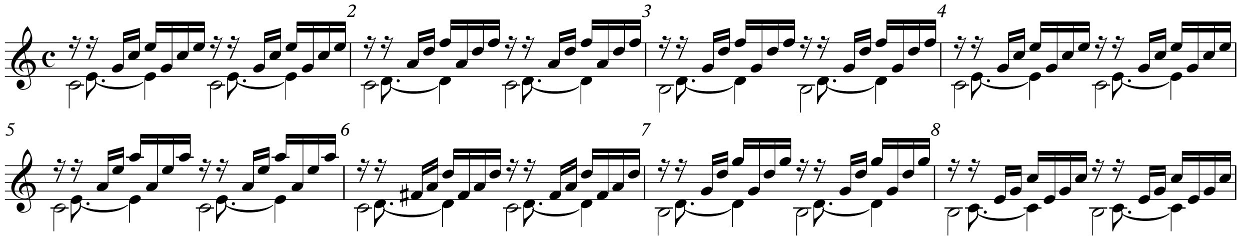


Fig. 4.8. Musical score of the sample "Prelude and Fugue No.1 in C major".

4.2.5 Towards automatic music transcription based on NMF technology

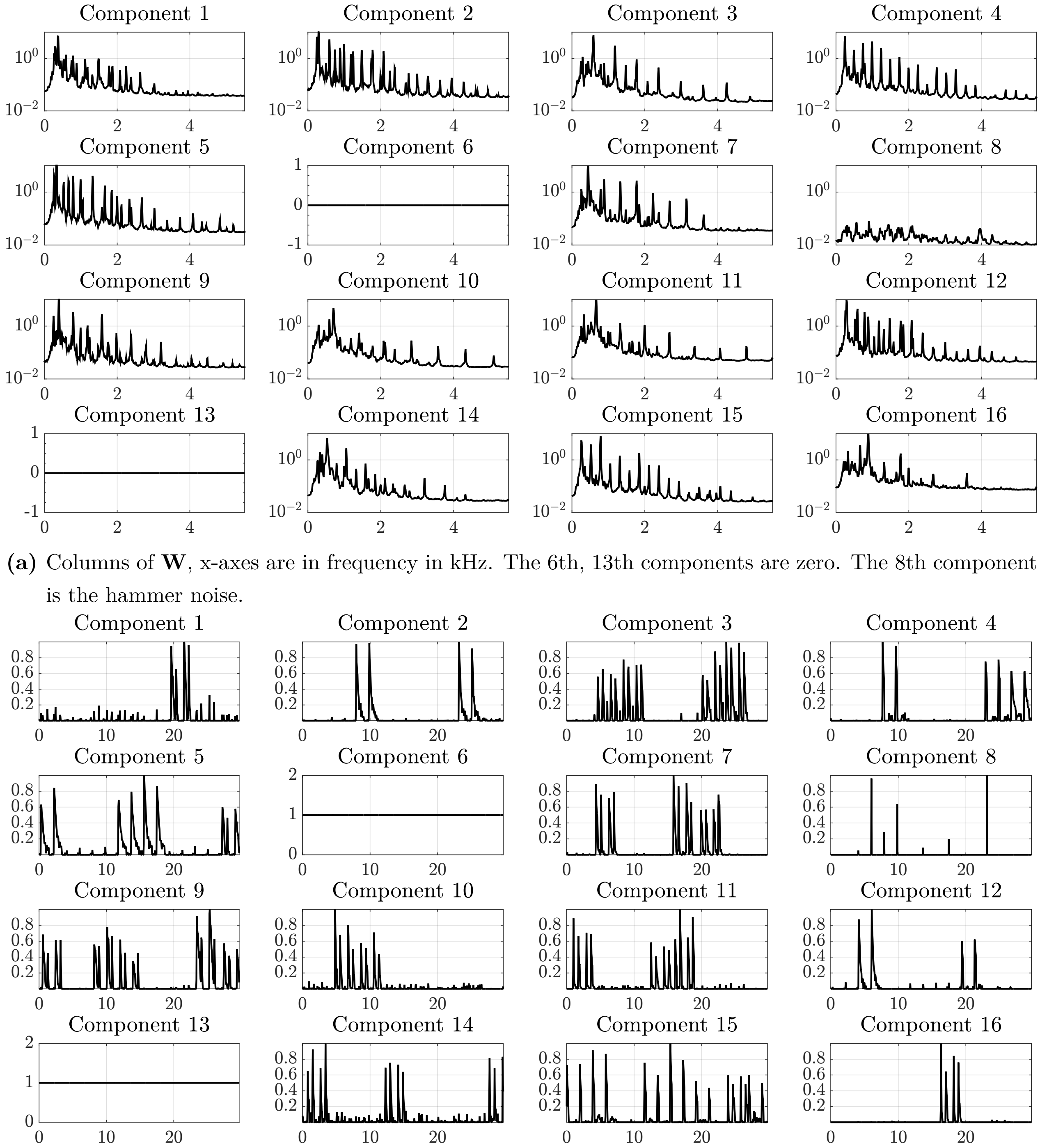
As a side note, Fig.4.10 only shows the activation timing of the note played on the piano. To further extract the duration and the decay of each note, we can make use of NuMF to be presented in §5 . The idea is to apply NuMF on the matrices $\mathbf{H}$ to obtain the activation and decay profile of each note; see Fig.5.10 for example. With the duration of the note identified, it is possible to build an automatic music transcription system on converting audio files into sheet music, fully based on NMF technology.

4.3 Chapter summary and perspective

We reviewed the audio BSS problem, and discussed how NMF can be used to tackle it. By solving a β-divergence NMF model, we demonstrated NMF can be used to decompose a single channel recording of a musical sequence, into components that represent each musical note.

Open problem 1: theoretical analysis of the model order selection In the experiments, we see that the minvol KL-NMF automatically set the overestimated components to zero. The theoretical

[1] https://www.youtube.com/watch?v=ZlbK5r5mBH4



(a) Columns of **W**, x-axes are in frequency in kHz. The 6th, 13th components are zero. The 8th component is the hammer noise.

(b) Rows of **H**, x-axes are in time in seconds. The 6th and 13th components are constant.

Fig. 4.9. Components **W** and **H** obtained with minvol KL-NMF with factorization rank $r = 16$ on the sample “Prelude and Fugue No.1 in C major".

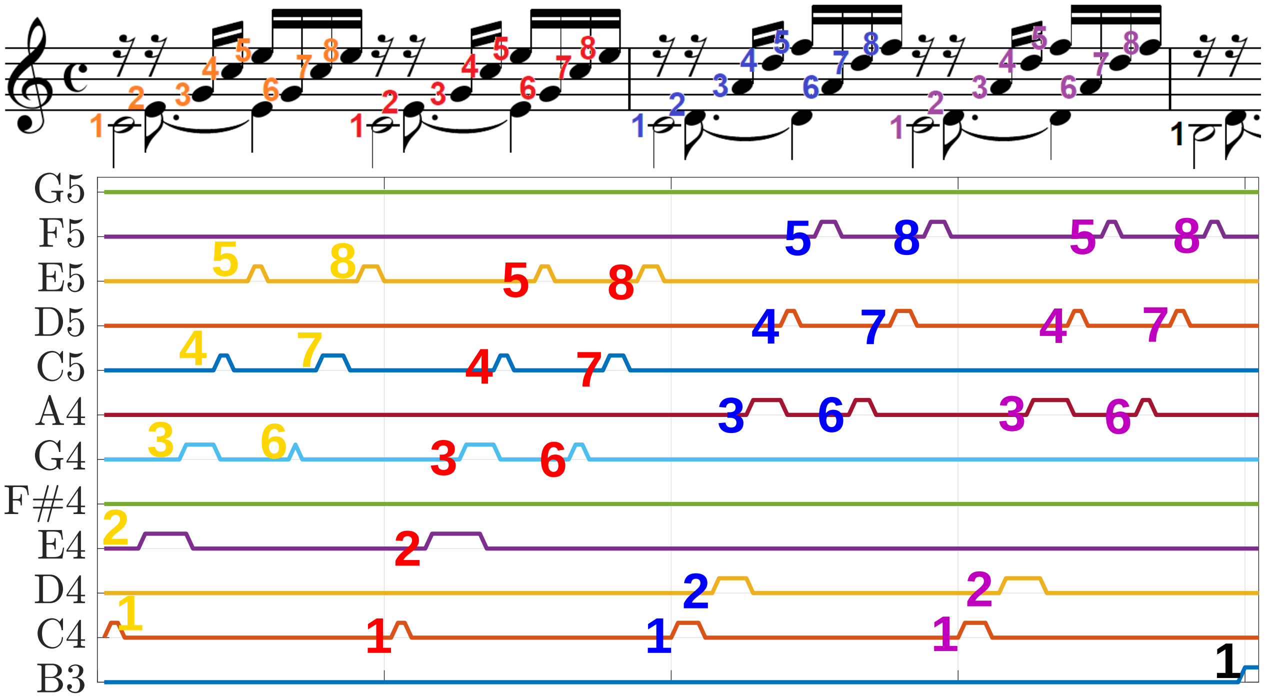


Fig. 4.10. Validation of the estimate sequence obtained with minvol KL-NMF with factorization rank $r = 16$ on the sample “Prelude and Fugue No.1 in C major". **Top**: The labeled music score of the first two measures of the music. **Bottom**: The corresponding activation (matrix $\mathbf{H}$ shown in Fig.4.9) after thresholding and permutation according to the pitch of the notes, determined by identifying the peak frequency location of the columns of matrix $\mathbf{W}$ shown in Fig.4.9. The result shows a perfect match between the score and the decomposition.

analysis of such phenomenon remains open.

Open problem 2: NMF extensions We can consider extending the NMF model to handle the violation of the assumptions listed in Table 4.2 . For example, as discussed in the experiment on the bell music, we can see that we can relax the assumption that each source (the temporal block) can be well-approximated by a rank-1 component. This leads to the study of Grouped NMF which consider the sub-patterns instead of the rank-1 components to be nonnegative.

Open problem 3: automatic music transcription fully based on NMF technology It will be interesting to combine minvol KL-NMF with NuMF to build an automatic music transcription system that can capture both the activation and the duration profile of each music note. Such system can be used to convert audio files into sheet music, and it is fully based on NMF technology.

5 NMF with unimodality: NuMF

> Nothing takes place in the world whose meaning is not that of some maximum or minimum.
>
> *Leonhard Euler*

In this chapter we introduce a new NMF model, namely, the Nonnegative Unimodal Matrix Factorization (NuMF). NuMF adds on top of NMF the unimodal condition on the factor matrix $\mathbf{W}$. This model finds application on nonnegative unmixing where the data exhibits unimodal structure.

Chapter organization We first give an introduction to NuMF and lay-down the foundation in §5.1 , where we characterize the unimodal property and show that NuMF is a nonconvex and block-nonconvex (on variable $\mathbf{W}$) problem. We then propose a simple but naive brute-force heuristics strategy to solve NuMF, which the strategy is then improved by accelerated gradient descent and multi-grid method. The multi-grid method acts as a dimension reduction step to reduce the dimension of the problem, and we show that the restriction operator preserves the unimodality in §5.2 . Then we present three preliminary results regarding the identifiability of NuMF in §5.3 under three special cases. We present empirical results concerning the algorithms and the theories on NuMF in §5.4 on synthetic and real datasets. Finally, we summarize this chapter in §5.5 and present some open problems.

Highlights of contributions We provide an efficient yet simple numeric algorithm to solve NuMF, based on the combination of multi-grid method and brute-force search. We justify the use of multi-grid method by proving that the restriction operator preserves the nonnegative unimodality of the vectors (Theorem 5.2.1) . We provide three preliminary results regarding the identifiability of NuMF in §5.3 under three special cases (Theorems 5.3.1, 5.3.2 and 5.3.3), and we give some insights on arriving the general identifiability of NuMF.

5.1 NuMF

NuMF adds on top of NMF a condition that the columns of $\mathbf{W}$ are *unimodal.*

Definition 5.1.1. ***(Monotonicity)*** *A vector $\mathbf{x} \in \mathbb{R}^m$ is called monotonic (also known as isotonic [88]) if $x_1 \leq x_2 \leq \cdots \leq x_m$ or $x_1 \geq x_2 \geq \cdots \geq x_m$. We call it strictly monotonic if the inequalities are all strict.*

Definition 5.1.2. ***(Unimodality and Nonnegative unimodality)*** *A vector $\mathbf{x} \in \mathbb{R}^m$ is called unimodal if there exists an integer $p \in [m]$ such that the tonicity of $\mathbf{x}$ changes at p from increasing to decreasing, i.e., $x_1 \leq x_2 \leq \cdots \leq x_p$ and $x_p \geq x_{p+1} \geq \cdots \geq x_m$. We let $\mathcal{U}^m$ be the set of unimodal vectors in $\mathbb{R}^m$, and let $\mathcal{U}^{m,p}$ be the set of vectors in $\mathcal{U}^m$ with tonicity change at p. A vector $\mathbf{x} \in \mathcal{U}^m$ is called nonnegative unimodal (Nu) if it is elementwise nonnegative, i.e.,*

$$0 \leq x_1 \leq x_2 \leq \cdots \leq x_p \quad \textit{and} \quad x_p \geq x_{p+1} \geq \cdots \geq x_m \geq 0, \;\textit{for some } p \in [m]. \tag{5.1}$$

We let $\mathcal{U}_+^m$ be the set of Nu vectors in $\mathbb{R}^m$, and let $\mathcal{U}_+^{m,p}$ be the set of Nu vectors in $\mathcal{U}_+^m$ with tonicity change at p. A matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ is Nu if all its columns are Nu, i.e., $\mathbf{x}_i \in \mathcal{U}_+^{m,p_i}$ for some p_i for all i. .

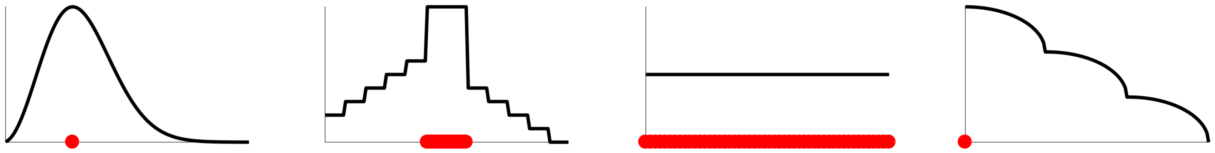


Fig. 5.1. Examples of vectors fulfilling Equation (5.1). The red dots in x-axis indicate the location of p's.

There are several remarks regarding the definitions.

- Equation (5.1) means the plot of vector $\mathbf{x}$ has a "single peak" on the top of the curve.
- In case of the equalities in (5.1), $\mathbf{x}$ may contain plateaus, and for such a Nu vector, the value of p is non-unique, see Fig.5.1 .
- The value of p is within $[m]$, for the last example in Fig.5.1, it has $p = 1$.
- Nu generalizes the notion of log-concavity [100]: a vector $\mathbf{x} \in \mathbb{R}^m_+$ is log-concave if

$$x_j^2 \geq x_{j-1}x_{j+1}, \text{ for all } j \in [2, m-1]. \tag{5.2}$$

  The condition (5.2) is called "log"-concave because applying a log-transform on (5.2) gives $2\log x_j \geq \log x_{j-1} + \log x_{j+1}$, and the relation $\dfrac{a_{j-1} + a_{j+1}}{2} \leq a_j$ for all $j \in [2, m-1]$ in a sequence $\{a_i\}$ is called concave. We only focus on Nu in this thesis as log-concavity is only a special case of Nu.

We now define NuMF.

Definition 5.1.3. ***(NuMF)*** *Given a matrix* $\mathbf{M} \in \mathbb{R}^{m \times n}_+$ *and a factorization rank* $r \in \mathbb{N}$, *find two matrices* $\mathbf{W} \in \mathbb{R}^{m \times r}$, $\mathbf{H} \in \mathbb{R}^{r \times n}$ *by solving the optimization problem*

$$\begin{array}{rl} \min\limits_{\mathbf{W},\mathbf{H}} & \frac{1}{2}\|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 + \lambda g(\mathbf{W}) \\ \textit{subject to} & \mathbf{w}_j \in \mathcal{U}^m_+ \textit{ for all } j \in [r], \\ & \mathbf{w}_j^\top \mathbf{1}_m = 1 \textit{ for all } j \in [r], \\ & \mathbf{h}^i \geq 0 \textit{ for all } i \in [r], \end{array} \tag{5.3}$$

where $g(\mathbf{W}) : \mathbb{R}^{m \times r} \to \mathbb{R}$ is a regularizer. We call (5.3) *fully-unimodal* if all rows of $\mathbf{H}$ are unimodal.

We now discuss Problem (5.3).

- The normalization constraint $\mathbf{w}_j^\top \mathbf{1}_m = 1$ balances the size of the columns of $\mathbf{W}$, and such instrumental constraint is for the purpose of handling the scaling ambiguity of the solution, see the discussion on Equation (1.7) in §1.4.4 . Similarly, we can put the constraint on $\mathbf{H}$ instead of $\mathbf{W}$ to achieve the same purpose. Here we put the normalization on columns $\mathbf{W}$ as the focus of NuMF is on the matrix $\mathbf{W}$.
- For NuMF, due to the property of Nu, the rank parameter r can be larger than n (the number of columns in $\mathbf{M}$), which is different from NMF, SNMF and minvol NMF. We come back to this issue in Theorems 5.3.1, 5.3.2, and we give an example illustrating this phenomenon in §5.4.

5.1.1 Motivations and applications of NuMF

Similar to minvol NMF, NuMF finds application in nonnegative unmixing problem. Here the specialty of NuMF is that it fits the data that exhibits unimodal structure, typically appeared in analytical chemistry such as the curve resolution problem, chromatography and flow injection analysis [16], and dynamics of infectious diseases [49].

GCMS Nu constraints is typical in the analysis of gas chromatography - mass spectrum (GCMS) data. The goal of GCMS is chemical unmixing and identification: given a chemical sample (data), identify the different substances presented in the sample; see Table 5.1 . We can see that the goals in Table 5.1 are highly similar to that in Table 3.1 on HU and in Table 4.1 on audio BSS.

Table 5.1: Three typical goals in GCMS.

	GCMS objective	Task in NuMF
(1)	Identify the number of substances.	Find r.
(2)	Obtain the spectral profile of the substances.	Find $\mathbf{W}$.
(3)	Identify the abundance of the substances.	Find $\mathbf{H}$.

We now briefly introduce the (simplified) mechanism of GCMS. Note that the purpose here is to introduce the minimum background on GCMS for the purpose of this thesis. We refer to the book [58] for a comprehensive treatment of GCMS.

Fig.5.2 shows the simplest setup of a CGMS system called headspace technique [58]. Consider the chemical sample containing two chemicals A and B in liquid state. The sample is heated up and the chemicals change from liquid to gas, the gas will leave the bottle and enter a very long tube called the analytical column. Due to the physical properties of the chemicals, A and B will have a different movement speed in the tube, and hence they will arrive at the sensor at different time. The sensors then produces two type of data: gas chromatography (GC) data and mass spectrometery (MS) data, according to the machinery setup. In GC data, the x-axis is the arrival time of the gas and the y-axis is the sensor respond. In MS data, the x-axis is the mass-to-charge ratio and the y-axis is the intensity. We stress that here the axes are not the main concern, the key idea here is the curves in the data correspond to individual chemicals, and we can identify the chemical presented in the sample by analyzing the data. As the axes are not important for the purpose of this thesis, from now on we do not explicitly label the axes, we just plot the axes as index of the vector.

Refer to Fig.5.2 , the GCMS data are typically formed as some conic combinations of Nu vectors, so it is justifiable to use NuMF for decomposition. In Fig.5.3 , we shows an example of real GCMS data on Belgian beers.

To learn the underlying vectors that generate the data, Nu constraints are added to the data model, which can be formulated as NuMF. As unimodal-shaped GCMS data often corresponds to a specific chemical, the Nu property helps to identify the chemicals presented in the sample. Without the unimodality, NMF alone may not be able to identify the chemicals as the decomposition may contain multiple peaks, we come back to this issue when we present the experimental result on CGMS data in §5.4 .

Dynamics of infectious diseases Another motivation of NuMF is that it can be used to model the dynamics of Nu time series, such as the dynamics of infectious diseases. Fig. 5.4 shows an example

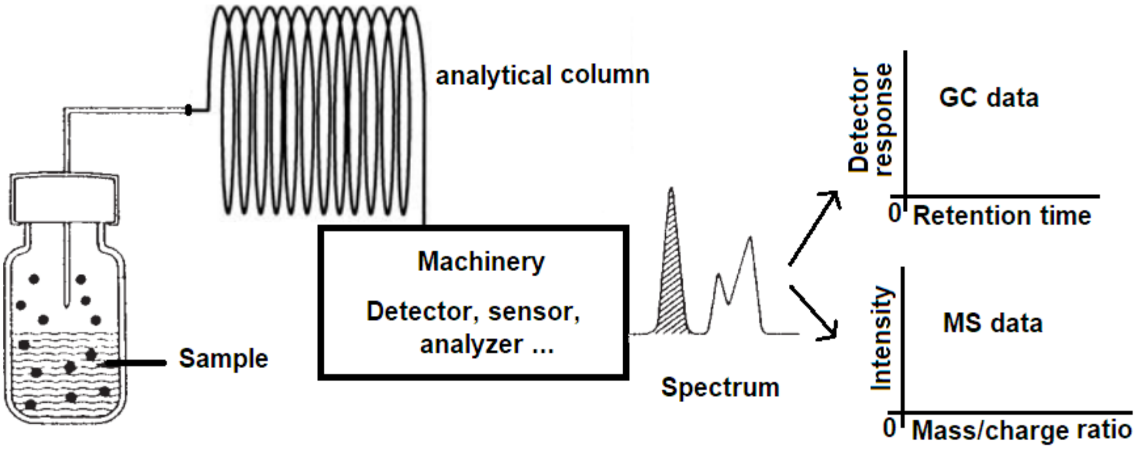


Fig. 5.2. The simplest setup of a CGMS system called headspace technique. Figure is modified from Fig. 2.16 of [58].

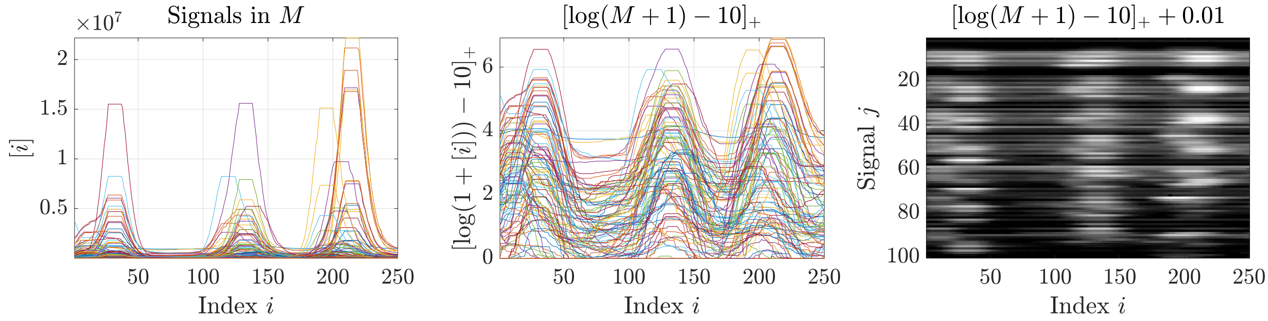


Fig. 5.3. A subset of CGMS data on Belgian beers [105], showing it is Nu.

on Measles incidence in London, the time curves are obviously Nu.

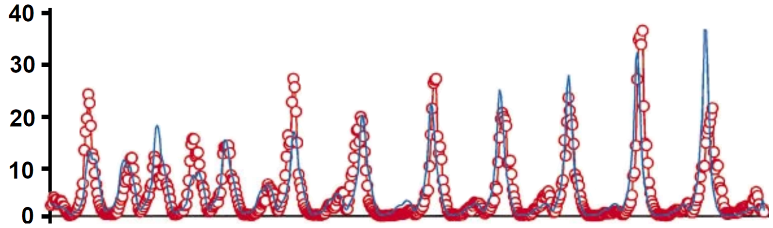


Fig. 5.4. Measles incidence in 2-week periods (in hundreds) in London from 1944 to 1965, showing it is Nu. Here x-axis is time (in week) and the y-axis is the number of incidence. Figure is reproduced from [49]

5.1.2 A characterization of the set of nonnegative unimodal vectors

The set $\mathcal{U}_+^{m,p}$ (Definition 5.1.2) is convex: $\mathbf{x}, \mathbf{y} \in \mathcal{U}_+^{m,p}$ implies $(1-\lambda)\mathbf{x} + \lambda\mathbf{y} \in \mathcal{U}_+^{m,p}$ for all $\lambda \in [0,1]$, but the set $\mathcal{U}_+^m$ is nonconvex for $m \geq 3$. For example, let $\mathbf{e}_i$ be the standard basis vector, then both $\mathbf{e}_i$ and $\mathbf{e}_j$ are unimodal, but the vector $(1-\lambda)\mathbf{e}_j + \lambda\mathbf{e}_j$ with $\lambda \in [0,1]$ is not, if $|j-i| \geq 2$. This example explains why the set $\mathcal{U}_+^m$ is nonconvex: the sets $\mathcal{U}_+^{m,i}$ with different i are disjoint, and the set $\mathcal{U}_+^m = \bigcup_i \mathcal{U}_+^{m,i}$ which is the union of disjoint convex sets, is nonconvex for $m \geq 3$.

When $m \geq 3$, the subproblem of solving $\mathbf{W}$ in NuMF (5.3), while fixing $\mathbf{H}$, is nonconvex, and hence NuMF problem is nonconvex and block-nonconvex.

In order to solve the nonconvex the subproblem on $\mathbf{W}$ in NuMF, we make use of the following convex characterization of the Nu set. Consider the union of two Nu sets: $\mathcal{U}_+^{m,p} \cup \mathcal{U}_+^{m,q}$, such set is describing the union of two systems of monic inequalities, and this union is a convex set if $|p-q| \leq 1$ (see Equation (5.4a) below). This motivates to work with the set $\mathcal{U}_+^{m,p} \cup \mathcal{U}_+^{m,p+1}$ (where p is known), instead of the general nonconvex set $\mathcal{U}_+^{m}$ (where p is unknown). The set $\mathcal{U}_+^{m,p} \cup \mathcal{U}_+^{m,p+1}$ admits a linear (and thereby convex) characterization:

$$\underbrace{\mathbf{x} \in \mathcal{U}_+^m}_{\text{“}\mathbf{x}\text{ is unimodal”}} \iff \exists p \text{ s.t. } \mathbf{x} \in \underbrace{\mathcal{U}_+^{m,p} \cup \mathcal{U}_+^{m,p+1}}_{\text{a convex set}} \overset{(5.1)}{\iff} \underbrace{\left\{\begin{array}{ccc} 0 & \leq & x_1 \\ x_1 & \leq & x_2 \\ & \vdots & \\ x_{p-1} & \leq & x_p \\ x_{p+1} & \geq & x_{p+2} \\ & \vdots & \\ x_{m-1} & \geq & x_m \\ x_m & \geq & 0 \end{array}\right.}_{\substack{\text{Union of two systems} \\ \text{of monic inequalities.}}} \iff \mathbf{U}_p \mathbf{x} \geq 0, \tag{5.4a}$$

so we have a matrix-form compact representation of the Nu membership of $\mathbf{x}$, where

$$\mathbf{U}_p = \left(\begin{array}{c|c} \underbrace{\begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}_{p \times p}}_{\mathbf{D}} & \mathbf{0}_{p \times (m-p)} \\ \hline \mathbf{0}_{(m-p) \times p} & \underbrace{\begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \\ & & & 1 \end{bmatrix}_{(m-p) \times (m-p)}}_{\mathbf{D}} \end{array}\right) \in \{-1, 0, 1\}^{m \times m}, \tag{5.4b}$$

is a matrix built by two first-order difference operators $\mathbf{D}$. By construction, the matrix $\mathbf{U}_p$ is block tri-diagonal and full rank. Being block tri-diagonal, the inverse of $\mathbf{U}_p$ is a block tridiagonal positive matrix. Such property becomes useful in solving NuMF, we comeback on this issue in §5.1.7 .

The index p plays a critical role in NuMF, as knowing p makes NuMF easier to solve, which is the main focus in the next subsection. Note that it is most likely that NuMF is a NP-hard problem.

5.1.3 Solving NuMF when p is known

NuMF essentially asks for searching r vectors in the set $\mathcal{U}_+^m$ to be the matrix $\mathbf{W}$. If the value of p of these columns in $\mathbf{W}$ are all known, NuMF Problem (5.3) can be solved by utilizing the convex characterization of the Nu set, i.e., Equation (5.4), to transform NuMF Problem (5.3) to the following problem

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2} \|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 + \lambda g(\mathbf{W}) \;\text{ s.t. }\; \mathbf{U}_{p_j} \mathbf{w}_j \geq 0, \; \mathbf{w}_j^\top \mathbf{1}_m = 1, \; \mathbf{h}^i \geq 0 \text{ for all } i, j \in [r]. \tag{5.5}$$

where p_j is the (known) index of the tonicity change of the optimal solution vector $\mathbf{w}_j^*$. The difference between Problems (5.5) and (5.3) is that the constraints on $\mathbf{W}$ in (5.5) are convex. Hence, if the regularizer g is a convex function in $\mathbf{W}$, the subproblem on $\mathbf{W}$ of Problem (5.5) is convex.

HALS discussed in §1.4.3 can be adopted to solve Problem (5.5). We can update $\mathbf{w}_i$ and $\mathbf{h}^i$ by solving the following two minimization subproblems

$$\underset{\mathbf{h}^i}{\operatorname{argmin}}\; Q(\mathbf{h}^i) = \frac{\|\mathbf{w}_i\|_2^2}{2}\|\mathbf{h}^i\|_2^2 - \langle \mathbf{M}_i^\top \mathbf{w}_i, \mathbf{h}^i\rangle \text{ s.t. } \mathbf{h}^i \geq 0, \tag{5.6a}$$

$$\underset{\mathbf{w}_i}{\operatorname{argmin}}\; Q(\mathbf{w}_i) = \frac{\|\mathbf{h}^i\|_2^2}{2}\|\mathbf{w}_i\|_2^2 - \langle \mathbf{M}_i(\mathbf{h}^i)^\top, \mathbf{w}_i\rangle + g(\mathbf{w}_i) \text{ s.t. } \mathbf{U}_{p_i}\mathbf{w}_i \geq 0, \;\; \mathbf{w}_i^\top \mathbf{1}_m = 1. \tag{5.6b}$$

Subproblem (5.6a) has closed-form solution given by the HALS Equation (1.4):

$$\mathbf{h} = \Big[\mathbf{M}_i^\top \mathbf{w}_i\Big]_+ \Big/ \|\mathbf{w}_i\|_2^2.$$

For subproblem (5.6b) , if $g(\mathbf{w}_i)$ is a quadratic function of $\mathbf{w}_i$, say $g(\mathbf{W}) = \det(\mathbf{W}^\top\mathbf{W})$, then the subproblem is a Linearly Constrained QP (LCQP). For example, $g(\mathbf{W}) = \det(\mathbf{W}^\top\mathbf{W})$, using Equation (2.6) ,

$$Q(\mathbf{w}_i) = \frac{1}{2}\mathbf{w}_i^\top \Big(\|\mathbf{h}^i\|_2^2\mathbf{I}_m + \lambda\gamma_i(\mathbf{I}_m - \mathbf{W}_{-i}(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})^\top\mathbf{W}_{-i}^\top)\Big)\mathbf{w}_i - \langle \mathbf{M}_i{\mathbf{h}^i}^\top, \mathbf{w}_i\rangle + c.$$

where $\gamma_i = \det(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})$ and $\mathbf{W}_{-i}$ denotes matrix $\mathbf{W}$ without $\mathbf{w}_i$. With such $Q(\mathbf{w}_i)$, subproblem (5.6b) has the form

$$\underset{\mathbf{x}}{\operatorname{argmin}}\; Q(\mathbf{x}) = \frac{1}{2}\langle \Theta\mathbf{x}, \mathbf{x}\rangle - \langle \theta, \mathbf{x}\rangle \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} \geq 0, \; \mathbf{x}^\top\mathbf{1} = 1, \tag{5.7a}$$

where

$$\boldsymbol{\Theta} = \|\mathbf{h}^i\|_2^2\mathbf{I}_m + \lambda\det(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})(\mathbf{I}_m - \mathbf{W}_{-i}(\mathbf{W}_{-i}^\top\mathbf{W}_{-i})^\top\mathbf{W}_{-i}^\top), \tag{5.7b}$$

$$\boldsymbol{\theta} = \mathbf{M}_i{\mathbf{h}^i}^\top, \tag{5.7c}$$

$$\mathbf{A} = \mathbf{U}_{p_i}, \tag{5.7d}$$

Problem (5.7) can be solved by Interior Point Method (IPM). However IPM is slow when the dimension m is large, and for practicality, we propose to solve Problem (5.7) by the Accelerated Projected Gradient (AccProjG) method, where we put the details in §5.1.7 .

5.1.4 Brute force approach to NuMF

Solving Problem (5.5) gives solution to Problem (5.3) when the values of p_j in the convex characterization (5.4) for all $\mathbf{w}_j$ are known. In general they are unknown and should be optimized. In this sense, NuMF is a problem with r integer variables $p_1, \dots, p_r$.

Definition 5.1.4. ***(NuMF)*** *Given a matrix* $\mathbf{M} \in \mathbb{R}_+^{m\times n}$ *and a factorization rank* $r \in \mathbb{N}$, *find two matrices* $\mathbf{W} \in \mathbb{R}^{m\times r}$, $\mathbf{H} \in \mathbb{R}^{r\times n}$, *and the indices* $p_1, \dots, p_r \in [m]$, *by solving the nonconvex optimization problem*

$$\min_{\substack{\mathbf{W},\mathbf{H}\\ p_1,\dots,p_j}} \frac{1}{2}\|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2 + \lambda g(\mathbf{W}) \;\; \textit{s.t.} \;\; \mathbf{U}_{p_j}\mathbf{w}_j \geq 0, \;\; \mathbf{w}_j^\top\mathbf{1}_m = 1 \;\; \mathbf{h}^i \geq 0 \; \textit{for all} \;\; i,j \in [r]. \tag{5.8}$$

As Problem (5.8) involves integer variables, it can be cast as a Mixed-Integer Programming (MIP) problem by introducing another set of binary variables that “turn on or off” of the structure $\mathbf{U}_{p_i}$. We propose to solve this problem using brute-force.

How to find p naively We now discuss a simple but naive way to find those p's, in which we will improve it in §5.2 using multi-grid. Consider the problem of finding p for a column in $\mathbf{W}$. A naive way to find p is *brute force*: solve Problem (5.7) for all possible values of p, and pick the one with the best fit, i.e., we solve

$$\min_{p\in[\![m]\!]} \left\{ \operatorname*{argmin}_{\mathbf{w}} \left\{ Q(\mathbf{w}) \text{ s.t. } \mathbf{U}_p\mathbf{w} \geq 0, \mathbf{w}^\top \mathbf{1}_m = 1 \right\} \right\}, \tag{5.9}$$

where $[\![m]\!]$ denotes the set of odd integers in $[m]$. Note that the search space of p is $[\![m]\!]$ instead of $[m]$ due to the constraint $\mathbf{U}_p\mathbf{x} \geq 0$ is representing the inclusion $\mathbf{x} \in \mathcal{U}_+^{m,p} \cup \mathcal{U}_+^{m,p+1}$ in the characterization (5.4).

Note that, the solution p of Problem (5.9) can be non-unique, see the discussion after Definition (5.1.2).

The complete algorithm is shown in Algorithm 3.

Algorithm 3 A simple but naive algorithm to solve NuMF

```
Input: A matrix $\mathbf{M} \in \mathbb{R}_+^{m\times n}$, parameters $hp \in \{1,2,3\}$ (extrapolation/projection option of $\mathbf{H}$).
Output: matrices $\mathbf{W}, \mathbf{H}$ that approximately solve (5.8).
Initialize $\mathbf{W}, \mathbf{H}$
for $k = 1, \dots$ until some criteria is satisfied do
    for $i \in [r]$ do
        Update $\mathbf{h}^i$ by the HALS Equation (1.4): $\mathbf{h} = \left[\mathbf{M}_i^\top \mathbf{w}_i\right]_+ \Big/ \|\mathbf{w}_i\|_2^2$.
        if $p_i$ is known then
            Update $\mathbf{w}_i$ by solving (5.7), using any LCQP solver or AccProjG (see §5.1.7)
        else
            Update $\mathbf{w}_i$ by solving (5.9), using any LCQP solver, or AccProjG (see §5.1.7)
        end if
    end for
end for
```

This algorithm works but is not practical for large m, as solving (5.9) requires to search for p among the odd integers in the set $[\![m]\!]$ with $\lfloor \frac{m}{2} \rfloor$ number of elements, which is expensive unless $\lfloor \frac{m}{2} \rfloor$ is sufficiently small. In fact, updating $\mathbf{W}$ in one iteration in Algorithm 3 if $p's$ are unknown requires

$$\mathcal{O}\left(r \times \lfloor \frac{m}{2} \rfloor \times t \right)$$

complexity, where t is the complexity of solving Problem (5.7a) to update one column of $\mathbf{W}$.

5.1.5 Practical ways to improve the brute force strategy

The brute force approach works efficiently when $\lfloor \frac{m}{2} \rfloor$ is small. When $\lfloor \frac{m}{2} \rfloor$ is large, there are two practical ways to improve the efficiency of the brute force strategy: guessing the peak position and dimension reduction. Both the two methods aim to reduce the size of the search space for p, and we discuss them below.

Guessing the peak position The first way to improve the brute force approach is to try to guess the value of p from the data, say using peak detection algorithm [34]. Fig.5.5 shows an example of

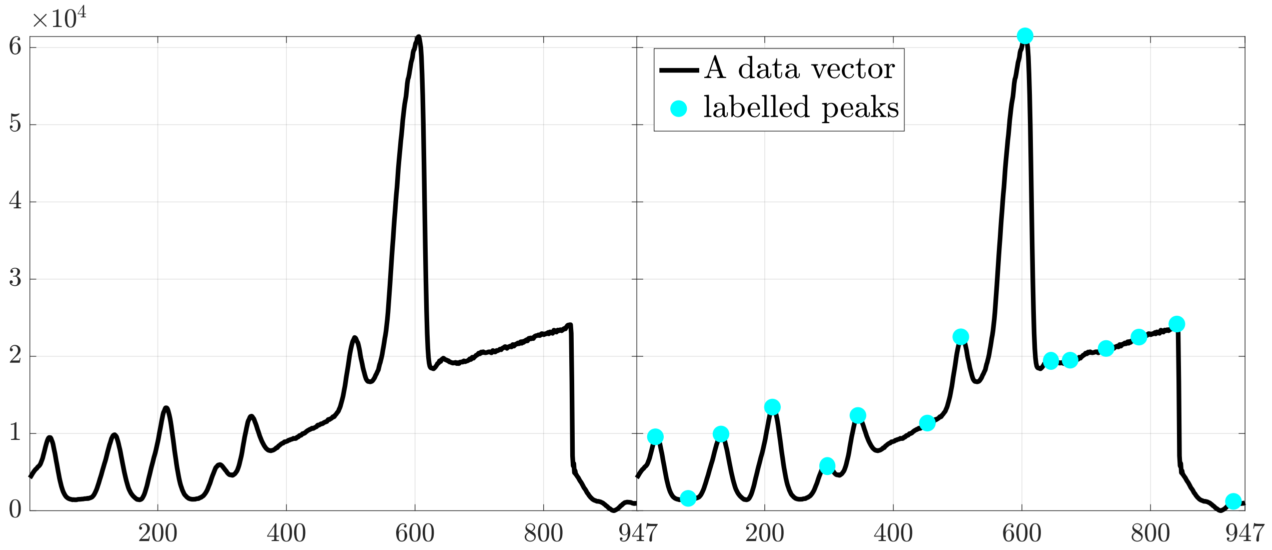


Fig. 5.5. Example of using peak detection on a data vector. Here $m = 947$ and the vector comes from a real GCMS data. Using the built-in peak detection `findpeaks` in MATLAB, 15 peaks are located. Using all the peak locations as the initial guess for p, the peak detection reduces the search space of p from $\lfloor \frac{947}{2} \rfloor = 473$ to 15.

using the built-in function in MATLAB to detect the peak location of a data vector, which reduces the search space for p from $\lfloor \frac{947}{2} \rfloor$ to 15.

Fig.5.5 shows the result on applying peak detection on a single vector, this trick can be used on a dataset (i.e., with many vectors). In this case, each data point (vector) in the dataset has its own set of peaks detected. We can cluster these peaks to get a rough estimate on the peak locations. See Fig.5.6 for an illustrative example.

It is important to note that, the effectiveness of the peak detection depends heavily on the quality of the data. Data properties such the peak intensity, peak shapes, degree of peak overlapping and amount of noise all affect the performance of peak detection.

Dimension reduction step Another way to improve the brute force approach is to use Dimension Reduction (DR) that reduces m to m', where $m' \ll m$. There are many DR techniques such as PCA and sampling, however both PCA and sampling do not preserve Nu, and thus they are not suitable for NuMF. The DR step we use is multi-grid, which preserves Nu (Theorem 5.2.1). We discuss the details of multi-grid in the next section.

In the remaining of this section, we 1) review some previous approaches on how to solve the NuMF, and 2) discus how to solve Problem (5.7a) efficiently with accelerated projected gradient.

5.1.6 Previous approaches on solving NuMF

We now review two approaches on solving Problem (5.9), i.e. the problem on solving on one single vector in $\mathbf{W}$.

Using dynamic programming To improve the efficiency of brute force approach to solve the Problem (5.9), an early attempt [16] used dynamic programming. It uses quantization of the entries of $\mathbf{x}$ to the closest value within a set $\{d_1, \dots, d_R\}$, and then performs a brute force search. For $\mathbf{x} \in \mathbb{R}^m$, the overall cost of such algorithm is $\mathcal{O}(R^2 m)$, where R is the number of elements of the set $\{d_1, \dots, d_R\}$. Note that for high accuracy of such approach, R has to be sufficiently large. Therefore,

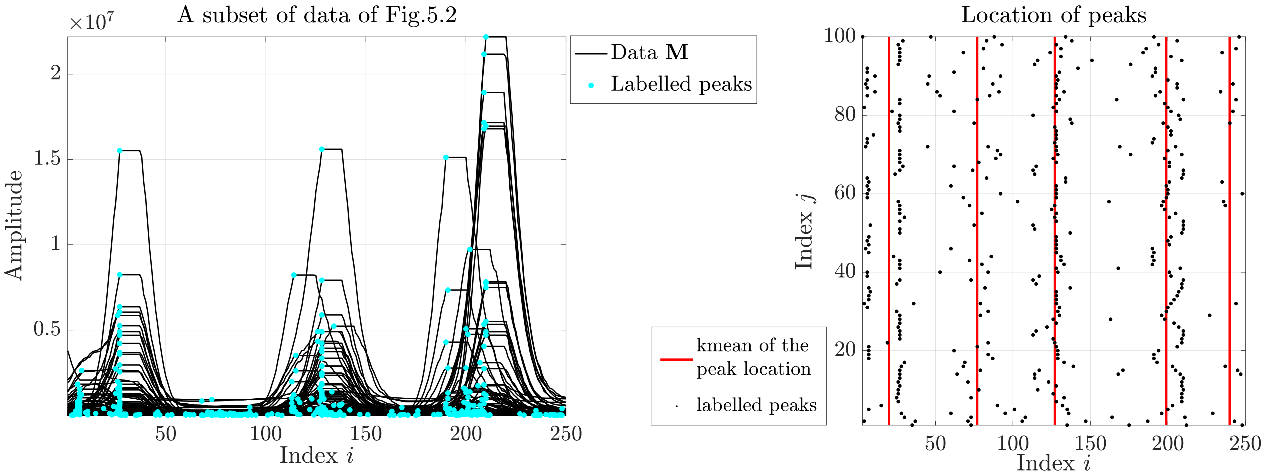


Fig. 5.6. Example of using peak detection on a dataset. Here we use the first 100 data points from the data presented in Fig.5.3, and here the matrix $\mathbf{M}$ is a 250-by-100 matrix. **Left**: we apply the peak detection on each data column $\mathbf{M}(:,j)$, and plot the data with the labeled peaks. **Right**: the peak locations plotted against the column index j. From the distribution of the points, we see 5 clusters. Using k-mean we get the estimate of the 5 peak locations plotted in red line. In this example, the peak detection reduces the search space of p from $\lfloor \frac{250}{2} \rfloor = 122$ to 5.

this approach can be inefficient when high accuracy is needed. Using such method to update $\mathbf{W}$ in one iteration has the complexity of $\mathcal{O}(rm\rfloor \times R^2 m)$.

Projected gradient based on two-branched isotonic projection Another approach is the projected gradient approach [19, section VI. B]. It is based on the isotonic projection algorithm [88] with a complexity in $\mathcal{O}(m)$. The idea is that, a Nu vector contains an increasing and a decreasing branch, and each branch is a monotonic sequence. Hence projection on to the Nu set $\mathcal{U}_+^m$ is equivalent to two projections onto two monotonic sequence, assuming p is known. Since in general p is not known, hence the same kind of brute force search discussed in §5.1.4 is required, and it takes $\mathcal{O}(r \times \lfloor \frac{m}{2} \rfloor \times m)$ of complexity for this method to update $\mathbf{W}$ in one iteration.

The two approaches both have the same overall complexity $\mathcal{O}(rm^2)$ for updating $\mathbf{W}$ in one iteration.

5.1.7 Solving Problem (5.7) using Accelerated Projected Gradient

We now present a method to solve Problem (5.7) using accelerated projected gradient (AccProjG). As a remark, this subsection is linked with §2.2.1 where we discussed how to solve minvol NMF subproblem (2.16) on a column of $\mathbf{H}$, here we solve an generalization of that problem.

First we restate Problem (5.7) :

$$(\mathcal{P}) \;:\; \underset{\mathbf{x}}{\operatorname{argmin}}\; Q(\mathbf{x}) = \frac{1}{2}\langle \Theta \mathbf{x}, \mathbf{x} \rangle - \langle \theta, \mathbf{x} \rangle \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} \geq 0,\; \mathbf{x}^\top \mathbf{1} = 1.$$

Directly applying AccProjG on Problem $(\mathcal{P})$ is inefficient as the constraint set $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} \geq 0, \mathbf{x}^\top \mathbf{1} = 1\}$ in $(\mathcal{P})$ is difficult to project. Hence, instead of solving $(\mathcal{P})$ using AccProjG directly, we transform $(\mathcal{P})$ to a problem that is easier to perform the projection. By design, the matrix $\mathbf{A}$ in $(\mathcal{P})$ is non-singular, so a change of variable $\mathbf{y} = \mathbf{A}\mathbf{x}$ gives

$$\operatorname*{argmin}_{\mathbf{y}} Q(\mathbf{y}) = \frac{1}{2}\langle \mathbf{Q}\mathbf{y}, \mathbf{y}\rangle - \langle \mathbf{p}, \mathbf{y}\rangle \quad \text{s.t.} \quad \mathbf{y} \geq 0,\ \mathbf{y}^\top \mathbf{b} = 1, \tag{5.10a}$$

$$\text{where} \quad \mathbf{Q} = \mathbf{A}^{-\top}\boldsymbol{\Theta}\mathbf{A}^{-1},\ \mathbf{p} = \mathbf{A}^{-1}\boldsymbol{\theta},\ \mathbf{b} = \mathbf{A}^{-1}\mathbf{1}, \tag{5.10b}$$

where $\mathbf{A}^{-\top}$ is the inverse of $\mathbf{A}^\top$. We solve Problem (5.10) by AccProjG (see Algorithm 4), the solution $\mathbf{y}^*$ is then converted back to $\mathbf{x}^*$ that solves Problem (5.7) by using the relation $\mathbf{y} = \mathbf{A}\mathbf{x}$.

Algorithm 4 Accelerated Projected Gradient (AccProjG) that solves Problem (5.10)

```
Input: $\mathbf{Q} \in \mathbb{R}^{m\times m}$, $\mathbf{p}$, $\mathbf{b}$
Output: Vector $\mathbf{y}$ that approximately solves (5.10).
Initialize $\hat{\mathbf{y}}_0 = \mathbf{y}_0 \in \mathbb{R}^m$
for $k = 1, \dots$ until some criteria is satisfied do
    $\mathbf{y}_k = P\Big(\hat{\mathbf{y}}_{k-1} - \frac{\mathbf{Q}\hat{\mathbf{y}}_{k-1} - \mathbf{p}}{\|\mathbf{Q}\|_2}\Big)$        % Projected Gradient Step
    $\hat{\mathbf{y}}_k = \mathbf{y}_k + \frac{k-1}{k+2}(\mathbf{y}_k - \mathbf{y}_{k-1})$        % Extrapolation step
end for
```

The key in Algorithm 4 is the projection operator P. Given a vector $\mathbf{z}$, $P(\mathbf{z})$ is defined as

$$P(\mathbf{z}) = \operatorname*{argmin}_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|_2^2 \ \text{ s.t. } \ \mathbf{x} \geq 0,\ \mathbf{x}^\top \mathbf{b} = 1. \tag{5.11}$$

This problem is the projection onto an irregular simplex described by the vector $\mathbf{b}$. By design, $\mathbf{b} \overset{(5.10b)}{=} \mathbf{A}^{-1}\mathbf{1} \overset{(5.7d)}{=} \mathbf{U}^{-1}\mathbf{1}$. As $\mathbf{U}$ is built by two first-order difference operator (see Equation (5.4b)), then $\mathbf{U}$ is block tri-diagonal and the inverse $\mathbf{U}^{-1}$ is a positive block tri-diagonal matrix, hence $\mathbf{b} > 0$ and Problem (5.11) satisfies the Slater's condition, i.e., the feasible region has a non-empty relative interior. The solution to Problem (5.11) can be derived by using the partial Lagrangian associated with the equality constraint:

$$L(\mathbf{x}, \nu) = \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|_2^2 + \nu(\mathbf{x}^\top \mathbf{b} - 1),$$

where the optimal $\mathbf{x}$, denoted as $\mathbf{x}^*$, can be obtained as

$$\begin{aligned} \mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}\geq 0} L(\mathbf{x},\nu) &= \operatorname*{argmin}_{\mathbf{x}\geq 0} \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|_2^2 + \nu\mathbf{x}^\top\mathbf{b} \\ &= \operatorname*{argmin}_{\mathbf{x}\geq 0} \sum_i^m \left(\frac{1}{2}(x_i - z_i)^2 + \nu b_i x_i\right). \end{aligned}$$

The terms $\frac{1}{2}(x_i - z_i)^2 + \nu b_i x_i$ is minimized for $x_i \geq 0$ when $x_i = [z_i - \nu b_i]_+$. Therefore, solution to Problem (5.11) reduces to the following hard-thresholding operator

$$\mathbf{x} = [\mathbf{z} - \nu\mathbf{b}]_+, \tag{5.12}$$

where the Lagrangian multiplier ν is the root of the following piece-wise linear equation

$$\sum_{i=1}^m \max\{0, z_i - \nu b_i\} b_i - 1 = 0, \tag{5.13}$$

with $\dfrac{z_i}{b_i}$ $(i \in [m])$ as the break points. Assume there are $K \leq m$ nonzero break points, Equation (5.13) can be solved in $\mathcal{O}(m + K\log m)$ based on sorting the break points. The over complexity of the projection step is hence in between $\mathcal{O}(m)$ and $\mathcal{O}(m\log m)$, based on the value K.

5.2 Multi-grid algorithms for solving NuMF

In this section, we use multi-grid method to speed up Algorithm 3. The idea is to form a problem in a coarse grid, then search p by solving Problem (5.9) in the coarse grid by brute force, after that we interpolate the solution on the coarse grid back to the fine grid, and use it as the initialization for solving the original NuMF Problem. In particular, The problem in a coarse grid is obtained by performing a *restriction* on the data. When the size of the problem in coarse grid is sufficiently small, we use the brute force approach to solve Problem (5.9) to find p. Then we use the p to solve the original NuMF problem; see Algorithm 5. In the subsequent sections, we discuss the details on each step in Algorithm 5. Then in the second half of this section, we give the justification of why multi-grid is used for NuMF: it preserves NU properties of the vectors, (Theorem 5.2.1).

Algorithm 5 Multi-grid NuMF

```
Input: $\mathbf{M} \in \mathbb{R}_+^{m\times n}$ (data matrix), $r$ (factorization rank)
Output: Matrices $\mathbf{W}, \mathbf{H}$ that approximately solve (5.8).
Initialize $\mathbf{W}_0 \in \mathbb{R}_+^{m\times r}, \mathbf{H}_0 \in \mathbb{R}_+^{r\times n}$
Perform restriction: $\mathbf{M}^{[N]} = \mathbf{R}_N \dots \mathbf{R}_1\mathbf{M},\ \mathbf{W}_0^{[N]} = \mathbf{R}_N \dots \mathbf{R}_1\mathbf{W}_0$.
Solve problem in coarse grid: $[\mathbf{W}^{[N]}, \mathbf{H}', \mathbf{p}^{[N]}]$ =NuMF($\mathbf{M}^{[N]}, \mathbf{W}_0^{[N]}, \mathbf{H}_0$) by Algorithm 3.
Perform interpolation: $[\mathbf{W}_0, \mathbf{p}_0]$ =Interpolation($\mathbf{W}^{[N]}, \mathbf{p}^{[N]}$).
Solve original problem $[\mathbf{W}, \mathbf{H}]$ =NuMF($\mathbf{M}, \mathbf{W}_0, \mathbf{H}', \mathbf{p}_0$) by Algorithm 3.
```

5.2.1 The restriction operator preserves Nu property

In this subsection we propose a restriction operator that preserves the Nu property.

Restriction To make the optimization problem small, instead of solving the origin problem (denoted by $\mathcal{P}_0$) in the original row-dimension m, we solve a new problem (denoted by $\mathcal{P}_1$) in the new row-dimension m_1. In the problem $\mathcal{P}_1$, instead of dealing with the objective function $\|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F^2$, we deal with the objective function $\|\mathbf{M}^{[1]} - \mathbf{W}^{[1]}\mathbf{H}\|_F^2$, where $\mathbf{M}^{[1]} = \mathbf{R}\mathbf{M}$ and $\mathbf{W}^{[1]} = \mathbf{R}\mathbf{W}$ have row-dimension m_1 with $m_1 < m$ and $\mathbf{R}$ is the restriction operator defined as follows.

Definition 5.2.1. *Restriction operator* $\mathbf{R} : \mathbb{R}_+^m \to \mathbb{R}_+^{m_1}$ *is defined as* $\mathbf{x} \rightarrowtail \mathbf{R}\mathbf{x}$*, where* $\mathbf{R} \in \mathbb{R}_+^{m\times m_1}$ *with* $m_1 < m$ *has the form of* (5.14)*. If the input is a matrix, the operator is defined column-wise, i.e., for a matrix* $\mathbf{X}$ *with* n *columns,* $\mathbf{R}\mathbf{X} := [\mathbf{R}\mathbf{x}_1 \dots \mathbf{R}\mathbf{x}_n]$.

There are many choices to construct $\mathbf{R}$, for simplicity, we stick with the following

$$\mathbf{R}(a,b) = \begin{bmatrix} a & b & & & & \\ & b & a & b & & \\ & & \ddots & \ddots & \ddots & \\ & & & b & a & b \\ & & & & b & a \end{bmatrix},\ a \geq 0, b \geq 0,\ a + 2b = 1. \tag{5.14}$$

For example

$$\mathbf{R}\left(\frac{1}{2},\frac{1}{4}\right) = \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & & & & & \\ & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & \\ & & & & & \frac{1}{4} & \frac{1}{2} \end{bmatrix}.$$

With $\mathbf{R}$ applied on the data, the problem $\mathcal{P}_1$ has a smaller size than problem $\mathcal{P}_0$, the cost of performing brute force search is reduced, from searching the odd integers in $[\![m]\!]$ to the odd integers in $[\![m_1]\!]$.

What we have discussed is the 1-layer grid method. In case m_1 is still large, we can repeat the whole process to obtain a new problem with smaller row-dimension. In the general N-level grid method, we have a problem $\mathcal{P}_N$ with a N-level restricted objective function $\|\mathbf{R}^{[N]}\mathbf{M} - \mathbf{R}^{[N]}\mathbf{W}\mathbf{H}\|_F^2$ where $\mathbf{R}^{[N]} = \mathbf{R}_N\mathbf{R}_{N-1}\ldots\mathbf{R}_1$. The search space on p of problem $\mathcal{P}_N$ is now $[\![m_N]\!]$ with $m_N \ll m$, where m_N is the row-dimension of $\mathbf{R}_N$

Remark 3. *In the above multi-grid process, we only consider the dimension reduction on the row-dimension, the column dimension is untouched.*

Multi-grid preserves Nu An important property that makes multi-grid applies to NuMF is that the restriction operators in the form of (5.14) preserve the Nu property. Fig.5.7 gives some examples showing that multi-grid approximation preserves unimodality. Formally, we have the following theorem.

Theorem 5.2.1. *Let* $\mathbf{x} \in \mathcal{U}_+^{m,p}$ *and* $\mathbf{R} \in \mathbb{R}^{m \times m_1}$ *with the structure defined in* (5.14). *Then* $\mathbf{y} = \mathbf{R}\mathbf{x} \in \mathcal{U}_+^{m_1,p_y}$ *with* $p_y \in \left\{\frac{p}{2}-1, \frac{p}{2}, \frac{p}{2}+1\right\}$.

Proof. The matrix $\mathbf{R}$ in the form of (5.14) can be decomposed as the sum of three matrices $\mathbf{R}(1), \mathbf{R}(2), \mathbf{R}(3)$ that:

- $\mathbf{R}(1)$ only contains the elements a in $\mathbf{R}$.
- $\mathbf{R}(2)$ only contains the elements b on the right of a in $\mathbf{R}$, i.e., the upper triangular part of $\mathbf{R}$
- $\mathbf{R}(3)$ only contains the elements b on the left of a in $\mathbf{R}$, i.e., the lower triangular part of $\mathbf{R}$

For the ease of understanding, below we give a simple illustrative example with $m = 5$ and $m_1 = 3$.

$$\mathbf{R} = \begin{bmatrix} a & b & & & \\ & b & a & b & \\ & & & b & a \end{bmatrix} = a\underbrace{\begin{bmatrix} 1 & 0 & & & \\ & 0 & 1 & 0 & \\ & & & 0 & 1 \end{bmatrix}}_{\mathbf{R}(1)} + b\underbrace{\begin{bmatrix} 0 & 1 & & & \\ & 0 & 0 & 1 & \\ & & & 0 & 0 \end{bmatrix}}_{\mathbf{R}(2)} + b\underbrace{\begin{bmatrix} 0 & 0 & & & \\ & 1 & 0 & 0 & \\ & & & 1 & 0 \end{bmatrix}}_{\mathbf{R}(3)},$$

The matrices $\mathbf{R}(i)$, $i = 1, 2, 3$ are sampling operators multiplied by a constant. These sampling operators are either picking the odd or even indices of $\mathbf{x}$ in the product $\mathbf{R}(i)\mathbf{x}$. As a sub-vector of a Nu vector is Nu, so the vectors $\mathbf{R}(i)\mathbf{x}$ are all Nu. The remaining task in the proof is to show the sum $\sum_{i=1}^3 \mathbf{R}(i)\mathbf{x}$ is Nu by using the characterization of Nu set in Equation (5.4): we show that p values of the vectors $\mathbf{R}(i)\mathbf{x}$ are integers that are at most 1 distance away from each other.

Without loss of generality, we assume p of $\mathbf{x}$ is even[1] , so for $\mathbf{R}(2)$ and $\mathbf{R}(3)$, which are sampling the even entries of the vector $\mathbf{x}$, the products $\mathbf{R}(2)\mathbf{x}$ and $\mathbf{R}(3)\mathbf{x}$ have p value equal to $\frac{p}{2}$, and their sum is hence Nu since they share the same p.

For $\mathbf{R}(1)$, it is easy to see that the product $\mathbf{R}(1)\mathbf{x}$ is a Nu vector with p value either $\frac{p}{2}-1$, $\frac{p}{2}$ or $\frac{p}{2}+1$. These values are all at most 1 distance away from $\frac{p}{2}$. □

We give an illustration of Theorem 5.2.1 in Fig.5.7 , where we showed the plots of restriction operator applied on three Nu vectors.

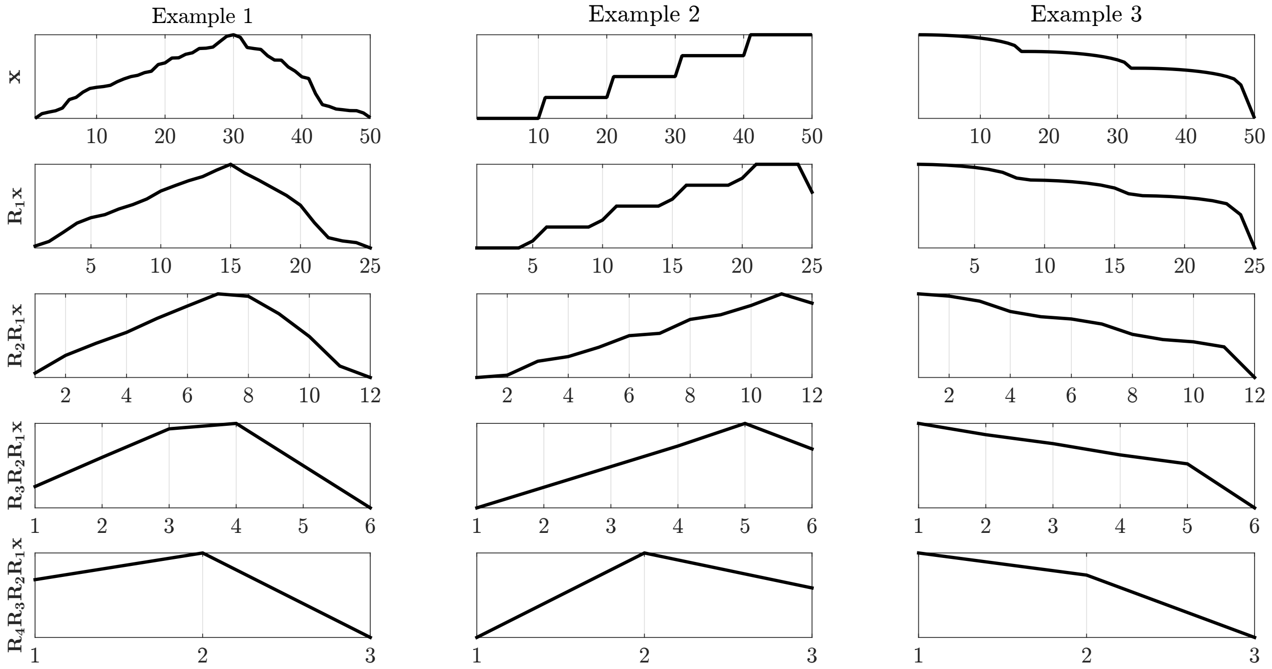


Fig. 5.7. Three Nu vectors and their multi-grid restrictions across 4 levels. All the vectors in this figure are Nu.

5.2.2 Interpolating the solution on coarse grid to fine grid

After the restriction has been applied on the data, we solve the problem on the coarse grid, and the solution is then interpolated back to the original fine grid by the interpolation operator.

Mathematically, after we solve the problem $\mathcal{P}_N$, we get $(\mathbf{W}^{[N]}, \mathbf{H}, \mathbf{p}^{[N]})$. For the matrix $\mathbf{H}$, as we do not perform restriction on the column-dimension, hence $\mathbf{H}$ obtained by solving problem $\mathcal{P}_N$ has the size $r \times n$, which can be used directly as initialization for solving problem $\mathcal{P}_0$. For the matrix $\mathbf{W}^{[N]}$, we can use the interpolation to obtain a matrix in the original dimension in the fine grid, which is then used as the initialization for solving problem $\mathcal{P}_0$. Finally, the vector $\mathbf{p}^{[N]}$ obtained by solving problem $\mathcal{P}_N$ correspond to the peak locations in $\mathbb{R}^{m_N}$, we interpolate $\mathbf{p}$ values to back to $\mathbb{R}^m$, denote as $\mathbf{p}_0$. Such $\mathbf{p}_0$ is then used to solve NuMF in Algorithm 3 , where now we have the information of p_i and hence no brute-force search is needed. That is, we do not enter step 10 in Algorithm 3 , which saves lot of computational resources.

[1] If p is odd, we can consider the vector $[x(1)\ \mathbf{x}]$, i.e., concatenate $x(1)$ to $\mathbf{x}$. Such concatenation does not change anything

Remark 4. *We give a remark concerning the implementation. It is important to note that, due to errors introduced by the restriction and interpolation processes, the vector* $\mathbf{p}_0$ *obtained from* $\mathbf{p}^{[N]}$ *may not be precisely accurate for solving NuMF in the original dimension. Hence, as a safeguard, we still perform a brute-force search for* p *values of NuMF in the original dimension, but we only search it in a small neighborhood of* $\mathbf{p}_0$*. The size of this neighborhood depends on the interpolation and restriction operators. Say, for* $\mathbf{R}$ *defined in Equation* (5.14)*, we search* ± 5 *of the values in* $\mathbf{p}_0$*.*

5.3 Preliminary identifiability results of NuMF

In this section we present the preliminary results on the identifiability of NuMF, i.e., when does solving NuMF gives a unique solution. We start by introducing the necessary terminologies, and then we present the lemmas and the identifiability results.

First we give the notation about the support of Nu vectors. The support of a vector $\mathbf{x} \in \mathcal{U}_+^m$ is denoted as $\text{supp}(\mathbf{x})$, defined as

$$\text{supp}(\mathbf{x}) = \{\ i \mid x_i \neq 0, i \in [m]\ \}.$$

The support of any Nu vector contains only a single close interval, hence we denote $\text{supp}(\mathbf{x}) = [a, b]$. That is, $x_i = 0$ for $i < a$ or $i > b$, and $x_i > 0$ for $a \leq i \leq b$. With the notion of support, we define the notion of disjoint between supports as follows.

Definition 5.3.1. ***(Disjoint, adjacent and strictly disjoint)*** *Consider two vectors* $\mathbf{x}, \mathbf{y} \in \mathcal{U}_+^m$ *with supp*$(\mathbf{x}) = [a_x, b_x]$ *and supp*$(\mathbf{y}) = [a_y, b_y]$*. The two vectors are called disjoint if supp*$(\mathbf{x}) \cap$ *supp*$(\mathbf{y}) = \varnothing$*. Next, on top of disjoint, we call the two vectors adjacent, if supp*$(\mathbf{x})$*, supp*$(\mathbf{y})$ *are adjacent to each other, i.e.,* $a_y = b_x + 1$*. Finally, on top of disjoint, we call the two vectors strictly disjoint if supp*$(\mathbf{x})$ *and supp*$(\mathbf{y})$ *are not adjacent to each other.*

Intuitively, two vectors that are strictly disjoint means that there is at least one zero in between their supports. This property in the strictly disjoint case becomes handy for proving the identifiability of NuMF as the conic combination of strictly disjoint vectors can not be Nu.

The opposite to disjoint is overlap. In short, there are two possibilities on overlap of supports: complete overlap and partial overlap. Concerning the preliminary identifiability result on NuMF, we do not go deep into the overlap cases.

5.3.1 When does conic combination preserves Nu?

To study the identifiability issue of NuMF, we have to first understand how conic combinations of Nu vectors behave. We now introduce a feasibility problem: given two linearly independent Nu vectors $\mathbf{x}, \mathbf{y} \in \mathcal{U}_+^m$, find $\alpha \in \mathbb{R}_{++}$ and $\beta \in \mathbb{R}_{++}$ such that the vector $\mathbf{z} := \alpha\mathbf{x} + \beta\mathbf{y}$ is Nu. This feasibility problem can be simplified as follows: as α, β are positive, we can set one of them as 1, and the simplified problem now reads

$$\begin{array}{rl} \text{Given} & \text{two linearly independent Nu vectors } \mathbf{x}, \mathbf{y} \in \mathcal{U}_+^m, \\ \text{Find} & \alpha \in \mathbb{R}_{++} \text{ such that } \mathbf{z} := \alpha\mathbf{x} + \mathbf{y} \text{ is Nu.} \end{array} \tag{5.15}$$

Regarding this feasibility problem, the results are as follows. If the support of $\mathbf{x}, \mathbf{y}$ are strictly disjoint, then Problem (5.15) has no solution, see Lemma 5.3.1 . If the support of $\mathbf{x}, \mathbf{y}$ are adjacent to each other, then feasibility of Problem (5.15) depends on the structure of the vectors, see lemmas 5.3.2 and 5.3.3 . We now discuss these lemmas.

Lemma 5.3.1. *If* $\mathbf{x}, \mathbf{y}$ *are strictly disjoint, then problem* (5.15) *has no solution.*

Proof. It is trivial based on definitions. □

For the adjacent case, without loss of generality let $\mathbf{x}, \mathbf{y} \in \mathcal{U}_+^m$ with $\text{supp}(\mathbf{x}) = [a, b]$, $\text{supp}(\mathbf{y}) = [c, d]$, $a \leq b < c \leq d$ and $c = b + 1$. An observation is that, if $\mathbf{x}$ is monotonically increasing within its support, then there exists a sufficiently small $\alpha > 0$ such that $\mathbf{z} = \alpha\mathbf{x} + \mathbf{y}$ is Nu, regardless of the toncity of $\mathbf{y}$. See Fig.5.8 for an example.

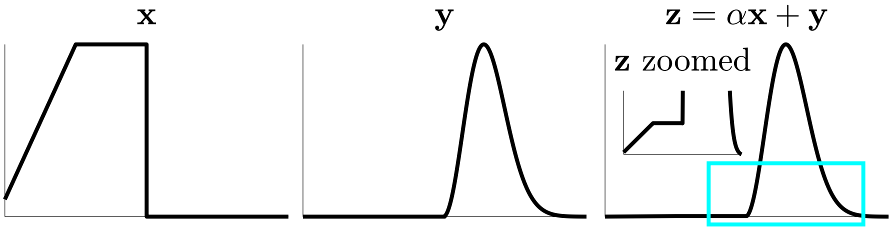


Fig. 5.8. Illustrative example of sum of two Nu vectors with adjacent support is Nu. Here $\text{supp}(\mathbf{x})$ is the first half of $[m]$ and $\text{supp}(\mathbf{y})$ is the second half of $[m]$. Note that $\mathbf{x}$ is unimodal in $[m]$ but monotonic in $\text{supp}(\mathbf{x})$.

The intuition here is that $\mathbf{x}$ is forming a "left hand sized tail" in $\mathbf{z}$. The same conclusion can be drawn on $\mathbf{y}$ on forming a "right hand sized tail" in $\mathbf{z}$. Such observation leads to Lemma 5.3.2 and 5.3.3 .

Lemma 5.3.2. *Let* $\mathbf{x}, \mathbf{y} \in \mathcal{U}_+^m$ *with* $\text{supp}(\mathbf{x}) = [a, b]$*,* $\text{supp}(\mathbf{y}) = [c, d]$*,* $a \leq b < c \leq d$ *and* $c = b + 1$*. Then if* $\mathbf{x}$ *is monotonically increasing in supp*$(\mathbf{x})$*, then* $\dfrac{y(c)}{x(b)}\mathbf{x} + \mathbf{y}$ *solves problem* (5.15)*.*

Proof. On the first case, we have $\text{supp}(\mathbf{z}) = \text{supp}(\mathbf{x}) \cup \text{supp}(\mathbf{y}) = [a, d]$ that $z(i) = \alpha x(i)$ if $i \in [a, b]$, and $z(i) = y(i)$ if $i \in [c, d]$. As $\mathbf{x}$ is monotonically increasing for $i \in [1, b]$, the sub-vector $\mathbf{z}(1 : b)$ is monotonically increasing. Regardless of the tonicity of $\mathbf{y}$, the vector $\mathbf{z}$ is Nu as long as $z(b) \leq z(c)$, or equivalently $\alpha x(b) \leq y(c)$. As $x(b) > 0$, we have $\alpha \leq \frac{y(c)}{x(b)}$. The vector $\frac{y(c)}{x(b)}\mathbf{x} + \mathbf{y}$ solves problem (5.15). □

Lemma 5.3.3. *Let* $\mathbf{x}, \mathbf{y} \in \mathcal{U}_+^m$ *with* $\text{supp}(\mathbf{x}) = [a, b]$*,* $\text{supp}(\mathbf{y}) = [c, d]$*,* $a \leq b < c \leq d$ *and* $c = b + 1$*. Problem* (5.15) *has no solution*

- *if* $\mathbf{x}$ *is monotonically decreasing in supp*$(\mathbf{x})$*, and there exists* $j \in [c, d - 1]$ *such that* $y(j) < y(j + 1)$*.*
- *if* $\mathbf{y}$ *is monotonically increasing in supp*$(\mathbf{y})$*, and there exists* $i \in [a+1, b]$ *such that* $x(i-1) > x(i)$*.*
- *if both* $\mathbf{x}$*,* $\mathbf{y}$ *are not monotonic in their support.*

Proof. The three cases means that there exists $i \in [a+1, b]$ such that $x(i-1) > x(i)$ and $j \in [c, d-1]$ such that $y(j) < y(j + 1)$. For all $\alpha > 0$ we have $z(i - 1) > z(i)$ and $z(j) < z(j + 1)$ for $j > i$. This violates Definition 5.1.2, so $\mathbf{z}$ is not Nu. □

We now discuss more about these lemmas. For Lemma 5.3.2 , if $\mathbf{y}$ is monotonically decreasing in $\text{supp}(\mathbf{y})$, then by symmetry $\mathbf{x} + \dfrac{y(c)}{x(b)}\mathbf{y}$ solves problem (5.15). For Lemma 5.3.3 , intuitively it means that there is a sub-vector $\mathbf{z}(i - 1, i, j, j + 1)$ in $\mathbf{z}$ containing a "V"-shape, so $\mathbf{z}$ is not unimodal.

5.3.2 Preliminary results on identifiability of NuMF

Here we give the preliminary identifiability result of NuMF on three special cases:

1. The strictly disjoint case for any value of r; see Theorem 5.3.1 .
2. The adjacent case for any value of r; see Theorem 5.3.2 .
3. On demixing two non-fully overlapped Nu vectors; see Theorem 5.3.3 .

Theorem 5.3.1. ***(Identifiability of NuMF: strictly disjoint case)*** *Let* $(\bar{\mathbf{W}}, \bar{\mathbf{H}})$ *be the matrices generating the data. Solving* (5.3) *recovers* $(\bar{\mathbf{W}}, \bar{\mathbf{H}})$ *if the followings are satisfied:*

1. $\bar{\mathbf{W}}$ *is Nu and all the columns have strictly disjoint support. Q*
2. $\bar{\mathbf{H}} \in \mathbb{R}_+^{r \times n}$ *has* $n \geq 1$, $\|\bar{\mathbf{h}}^i\|_\infty > 0$ *for* $i \in [r]$.

Proof. The proof is trivial, we include it here for completeness. Assume there is another solution $(\mathbf{W}^\#, \mathbf{H}^\#)$ that solves the NuMF. The columns $\bar{\mathbf{w}}_j$ contribute in $\mathbf{M}$ a series of disjoint unimodal components. For the solution $\mathbf{W}^*\mathbf{H}^*$ to fit $\mathbf{M}$, each column of $\mathbf{W}^*$ has to fit each of these disjoint component in $\mathbf{M}$, and hence $\mathbf{W}^*$ recovers $\bar{\mathbf{W}}$, subject to permutation and scaling. Thereby, we have $\mathbf{H}^*$ recovers $\bar{\mathbf{H}}$. □

Note that the first assumption in Theorem 5.3.1 seems very strong, but in fact many CGMS data fulfill this assumption: when the physical properties of the chemicals in the sample are very distinct, say their movement speeds in the gas phase are highly different, then there will be a huge gap between their Nu curves. Furthermore, note that the theorem holds even for any n even for $n = 1$ or $r \geq n$, which is not common in other NMF models (e.g. SNMF, minvol NMF) that assume $r \ll \min\{m, n\}$.

Now we move to the adjacent case.

Theorem 5.3.2. ***(Identifiability of NuMF: adjacent case)*** *Let* $(\bar{\mathbf{W}}, \bar{\mathbf{H}})$ *be the matrices generating the data. Solving* (5.3) *recovers* $(\bar{\mathbf{W}}, \bar{\mathbf{H}})$ *if the followings are satisfied:*

1. $\bar{\mathbf{W}}$ *is Nu and adjacent.*
2. $\bar{\mathbf{H}} \in \mathbb{R}_+^{r \times n}$ *has* $n \geq 1$, $\|\bar{\mathbf{h}}^i\|_\infty > 0$ *for* $i \in [r]$.
3. *Independent sensing: for each index pair* (j_1, j_2), $j_1, j_2 \in [r]$, $j_1 \neq j_2$ *such that the vectors* $\bar{\mathbf{w}}_{j_1}, \bar{\mathbf{w}}_{j_2}$ *satisfy the condition of the vectors* $\mathbf{x}, \mathbf{y}$ *in Lemma 5.3.2 , the* j_1, j_2 *rows of* $\mathbf{H}$ *contains a positive diagonal block* $\bar{\mathbf{D}}$.

Proof. By assumption 3 in the theorem, those $\bar{\mathbf{w}}_{j_1}, \bar{\mathbf{w}}_{j_2}$ satisfying Lemma 5.3.2 will have to appear separately in the data. So for $\mathbf{W}^*$ will have to recover them to for fitting the data $\mathbf{M}$. Thereby, we have $\mathbf{H}^*$ recovers $\bar{\mathbf{H}}$. Note that the fitting is subject to permutation and scaling, which does not change anything. □

Remark 5. *We give a toy example to illustrate the importance of assumption 3 in Theorem 5.3.2. Let $m = 6$ and $r = 3$ and we have $\bar{\mathbf{W}}$ as follows with a few additional matrices:*

$$\bar{\mathbf{W}} = \begin{bmatrix} 1 & & \\ 1 & & \\ & 1 & \\ & 1 & \\ & & 1 \\ & & 1 \end{bmatrix}, \quad \mathbf{E}_1 = 1.5 \begin{bmatrix} 1 & \\ 1 & \\ & 1 \\ & 1 \\ 1 & \\ 1 & \end{bmatrix}, \quad \mathbf{B}_1 = \begin{bmatrix} 1 & & \\ 1 & & \\ 1 & & \\ & 1 & \\ & 1 & \\ & & 1 \end{bmatrix},$$

$$\mathbf{C}_1 = \begin{bmatrix} 1.5 \\ 1.5 \\ 1.5 \end{bmatrix}, \qquad \mathbf{C}_2 = \begin{bmatrix} 1.5 & 1.5 \\ 1.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix}, \quad \mathbf{C}_3 = \begin{bmatrix} 1.5 & \\ & 1.5 \\ 1.5 & \end{bmatrix},$$

where an empty slot represents zero.

Here the pairs $(\bar{\mathbf{w}}_1, \bar{\mathbf{w}}_2)$, $(\bar{\mathbf{w}}_2, \bar{\mathbf{w}}_3)$ satisfy Lemma 5.3.2. If assumption 3 is not satisfied, say $\bar{\mathbf{H}}$ has the form of $\mathbf{C}_1$ or $\mathbf{C}_2$, then $\mathbf{M}$ contains copies of the vector $[1\,1\,1\,1\,1\,1]^\top$ (multiplied by 1.5). In this case, the problem is not identifiable as $\mathbf{B}_1$ is a feasible solution for $\mathbf{W}^*$. Note that there does not exists a full rank matrix $\mathbf{Q} \in \mathbb{R}^{3\times 3}$ such that $\bar{\mathbf{W}} = \mathbf{B}_1\mathbf{Q}$.

If assumption 3 is satisfied, say $\bar{\mathbf{H}}$ has the form $\mathbf{C}_3$, then the data $\mathbf{M}$ has the form of $\mathbf{E}_1$. In this case, the problem is identifiable as $\mathbf{W}^*$ can only have the form of $\bar{\mathbf{W}}$ (subject to permutation and scaling).

We now present the last preliminary result on general identifiability for NuMF with $\mathbf{W}$ limited to only two columns. This result is general in the sense that it includes the overlapped vectors which is not addressed in the two previous theorems. This result is preliminary in the sense that $r = 2$, unlike the two previous theorems that work for any r.

Theorem 5.3.3. ***(On demixing two non-fully overlapped Nu vectors)*** *Given two Nu vectors $\mathbf{x}, \mathbf{y}$ in $\mathcal{U}_+^m$ that supp$(\mathbf{x}) \not\subseteq$ supp$(\mathbf{y})$ and supp$(\mathbf{x}) \not\supseteq$ supp$(\mathbf{y})$. Suppose $\mathbf{x}, \mathbf{y}$ can be generated by two non-zero Nu vectors $\mathbf{u}$, $\mathbf{v}$ as*

$$\mathbf{x} = a\mathbf{u} + b\mathbf{v} \quad \textit{and} \quad \mathbf{y} = c\mathbf{u} + d\mathbf{v} \tag{5.16}$$

with nonnegative coefficients a, b, c, d. Then the only possibilities for (5.16) are either $\mathbf{u} = \mathbf{x}$, $\mathbf{v} = \mathbf{y}$ or $\mathbf{u} = \mathbf{y}$, $\mathbf{v} = \mathbf{x}$.

Proof. First without loss of generality we assume $\mathbf{u} \neq \mathbf{v}$ otherwise it is trivial. Let $\mathbf{X} = \mathbf{U}\mathbf{Q}$, where $\mathbf{X} := [\mathbf{x}, \mathbf{y}]$, $\mathbf{U} := [\mathbf{u}, \mathbf{v}]$ and $\mathbf{Q} := \begin{bmatrix} a & c \\ b & d \end{bmatrix}$. As a, b, c, d are all nonnegative hence $\mathbf{Q} \geq 0$.

The conditions that $\mathbf{x}, \mathbf{y}$ are Nu with supp$(\mathbf{x}) \not\subseteq$ supp$(\mathbf{y})$ and supp$(\mathbf{x}) \not\supseteq$ supp$(\mathbf{y})$ imply that $\mathbf{x} \neq 0, \mathbf{y} \neq 0$, $\mathbf{x} \neq \mathbf{y}$ and

$$\text{supp}(\mathbf{x}) \not\supseteq \text{supp}(\mathbf{y}) \implies \exists i^* \in [m] \text{ s.t. } x_{i^*} > 0, y_{i^*} = 0, \tag{5.17a}$$

$$\text{supp}(\mathbf{y}) \not\supseteq \text{supp}(\mathbf{x}) \implies \exists j^* \in [m] \text{ s.t. } y_{j^*} > 0, x_{j^*} = 0. \tag{5.17b}$$

Then $\mathbf{x} \neq \mathbf{y}$ and $\mathbf{u} \neq \mathbf{v}$ imply $\mathbf{X}, \mathbf{U}, \mathbf{Q}$ are all rank-2, hence

$$\mathbf{U} = \mathbf{X}\mathbf{Q}^{-1} = \mathbf{X} \begin{bmatrix} d & -c \\ -b & a \end{bmatrix} \frac{1}{ad - bc}, \quad ad - bc \neq 0. \tag{5.18}$$

Put the indices i^*, j^* in (5.17a),(5.17b) into (5.18), together with the fact that $\mathbf{x}, \mathbf{y}, \mathbf{u}, \mathbf{v}$ are nonnegative give $\mathbf{Q}^{-1} \geq 0$.

Lastly both $\mathbf{Q} \geq 0$ and $\mathbf{Q}^{-1} \geq 0$ imply $\mathbf{Q}$ is a permutation of a positive diagonal matrix [29], where here the diagonal matrix here is the identity matrix. □

It is trivial that, by making use of Theorem 5.3.3 , we immediately arrive at the identifiability of NuMF, for the case $r = 2$, under the conditions that the columns of $\mathbf{W}$ are Nu, and $\mathbf{H}$ is nonnegative. Notice that here we do not have condition on the interactions between the supports of the columns of $\mathbf{W}$. In general, generalizing Theorem 5.3.3 to any value of r gives the general identifiability of NuMF. However, no result is obtained so far.

The three theorems presented so far are addressing the identifiability of NuMF from two angles: 1) the number of columns in $\mathbf{W}$, 2) how the supports of these columns interact with each other. Neither of the three theorems is complete. A complete theorem on identifiability of NuMF should address all possible interaction between supports of the columns in $\mathbf{W}$ for any value of r, and this is a topic of further research.

5.4 Preliminary experimental results of NuMF

In this section, we present numerical experiments of NuMF on two synthetic datasets. We set $\lambda = 0$ (no regularization) unless it is specified. Note that in real world application, because of the present of noise, real world data is not exactly Nu, but it will be close enough if the noise is small.

Simple toy example 1: three overlapped Nu vectors We consider a toy example to illustrate how Nu property helps to recover the ground truth solution, and how multi-grid improves convergence. Let $(m, n, r) = (100, 6, 3)$. We construct a Nu matrix $\mathbf{W}$ such that the second component overlap with the other two components. Let $c = \frac{1}{\sqrt{2}}$ and $\epsilon = 0.001$, the matrix $\mathbf{H}$ is constructed as

$$\mathbf{H} = \begin{bmatrix} c+\epsilon & 1-c-\epsilon & c+\epsilon & 1-c-\epsilon & 0 & 0 \\ 1-c-\epsilon & c+\epsilon & 0 & 0 & c+\epsilon & 1-c-\epsilon \\ 0 & 0 & 1-c-\epsilon & c+\epsilon & 1-c-\epsilon & c+\epsilon \end{bmatrix}.$$

which is the Equation (2.9) that fulfills the SSC condition described in §2.1.5 .

Using the $\mathbf{W}$ and $\mathbf{H}$ we construct the data matrix $\mathbf{M} = \mathbf{WH}$. The NuMF problem with $r = 3$ is then solved with multi-grid method with 1 and 2 layers. For comparisons, we also solve the problem with no grid. The result is shown in Fig.5.9 , indicating that multi-grid approach can significantly speed up the convergence of the algorithm. We do not compare the result with other algorithms mentioned in §5.1.6 as they have similar complexities with the method with no grid in Fig.5.9 .

Simple toy example 2: with $r > n$ In this example, we illustrate a special property of NuMF that it is possible for $r > n$. Here the data $\mathbf{M}$ is a 295-by-1 matrix. Such matrix is in fact a column of the $\mathbf{H}$ matrix obtained by decomposing an audio spectrogram of "Mary had a little lamb", discussed in §4 . We will discuss the application of this specific NuMF in §4 . In this $\mathbf{M}$, we have $n = 1$ and $r = 2$. NuMF is able to decompose the data $\mathbf{M}$ into two Nu vectors, as shown in Fig.5.10 . Notice that here $r = 2 > 1 = n$, which is not possible for NMF, SNMF and minvol NMF. This example showed that, under the Nu property, it is possible for $r > n$, as long as these r components can be distributed within the interval $[1, 2, \ldots, m]$.

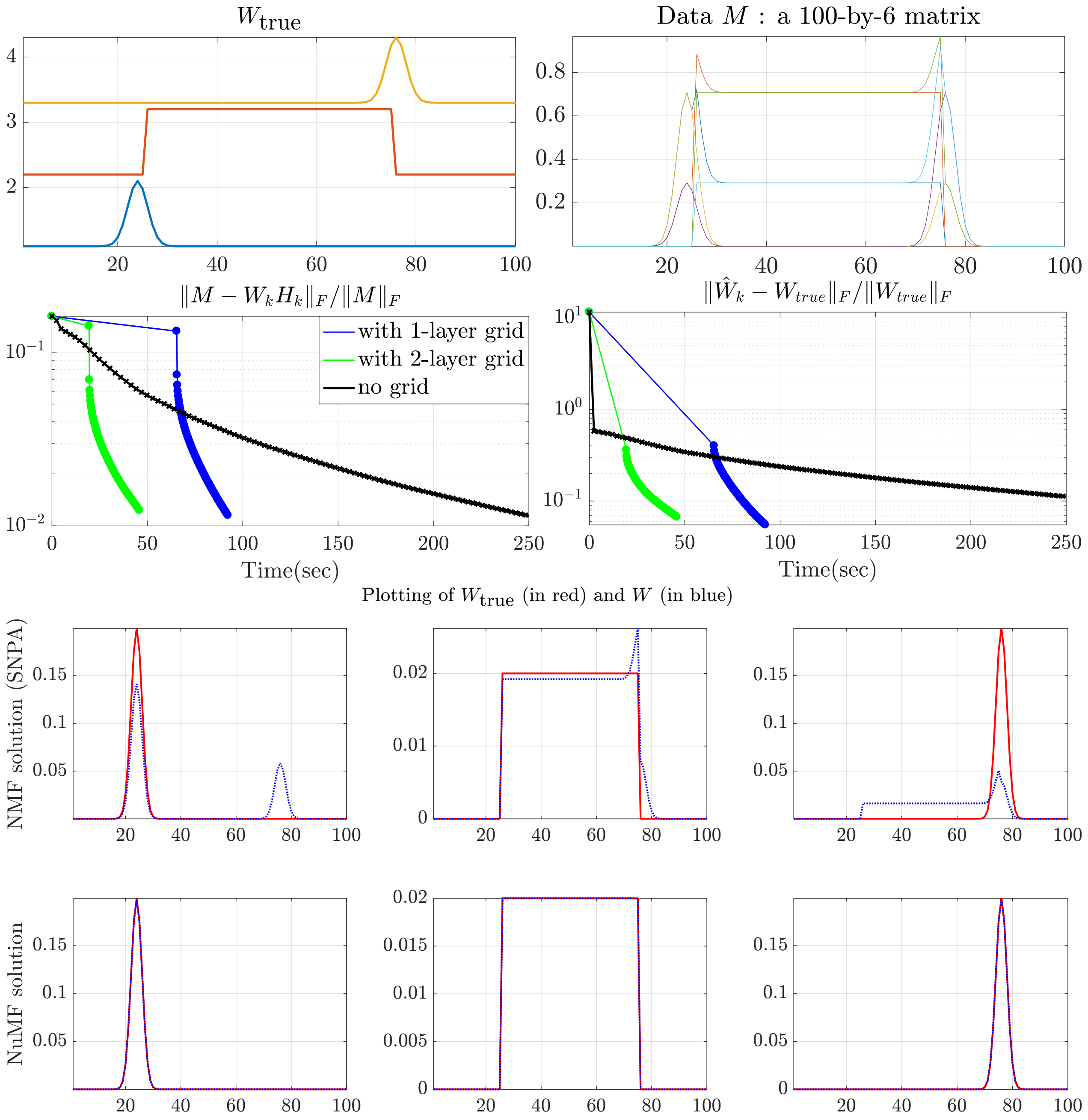


Fig. 5.9. Results of NuMF on the toy example 1: three overlapped Nu vectors. **Top row**: The matrix W_{true} and **M**. **Mid row**: results of the algorithms plotted against time. All algorithms run 100 iterations with SNPA initialization. For algorithms with multi-grid, the computational time taken on the coarse grid are also taken into account, as reflected by the time gap between time zero and the first dot of the curves. The result show that multi-grid can significantly improve the convergence speed of the method. **Bottom rows**: The plot of **W** (scaled), where red curve denotes the ground truth and the blue curve denotes the estimate. The result show that NuMF can identify the solution correctly while SNMF (obtained by SNPA) cannot.

On real dataset: GCMS data of Belgian beer We now apply NuMF on the Gas chromatography–mass spectrometry (GCMS) data of Belgian beers of Fig.5.3 , the result is shown in Fig.5.11. For comparisons, we also plot the results obtained from NMF and SNMF model. As expected, NuMF can decompose the data into individual Nu components, while NMF and SNMF cannot, as some of

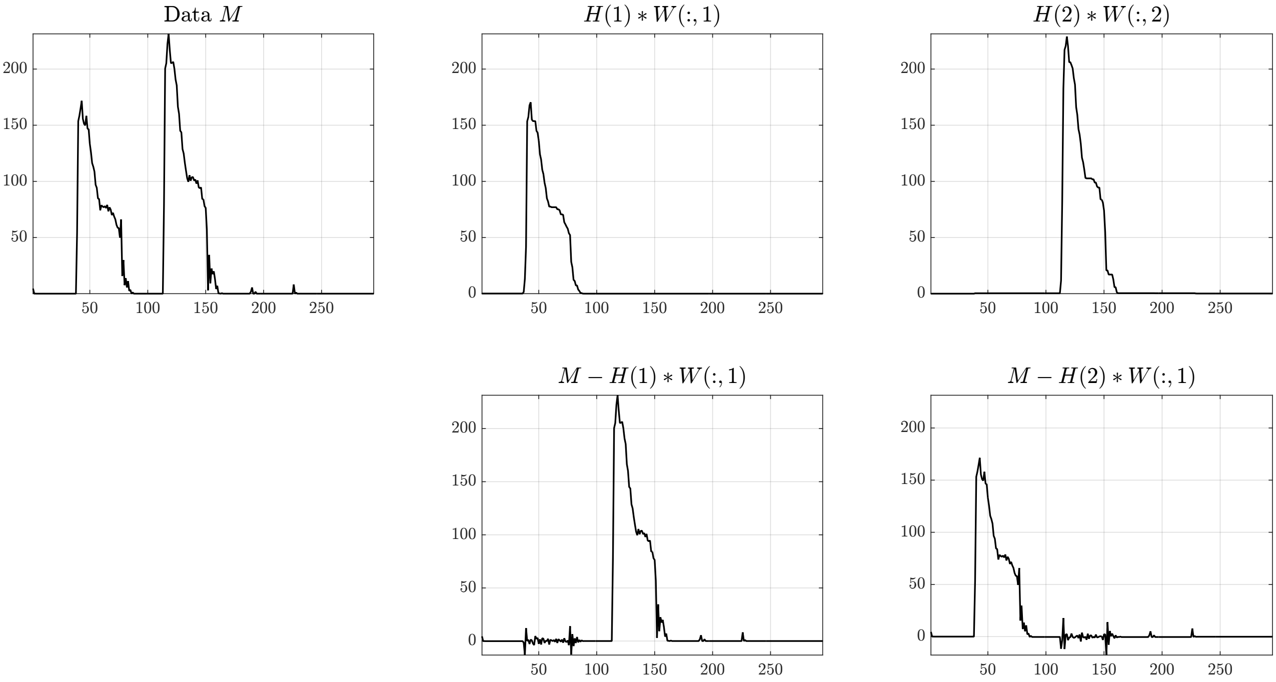


Fig. 5.10. Results of NuMF on the toy example 2: with $r = 2 > 1 = n$. Here NuMF can be used to decompose a vector into Nu components. This example also empirically verifies Theorem 5.3.1 , illustrated by the small residue of $M - H(i) * W(:,i)$ in the support $\text{supp}(W(:,i))$.

their components are highly mixed with multiple peaks.

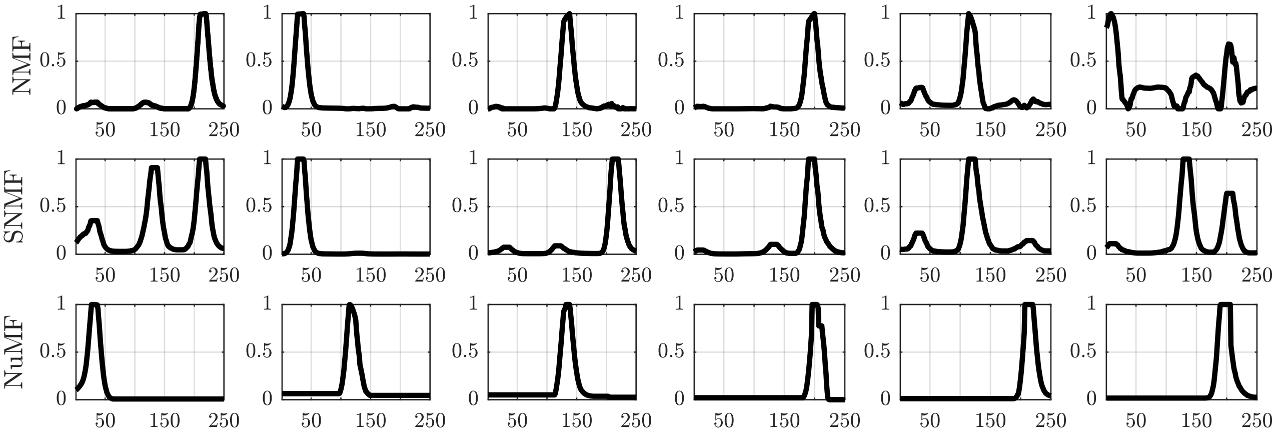


Fig. 5.11. Decomposing GCMS data of Belgian beer with $r = 7$. The plots are columns of $\mathbf{W}$ obtained by three methods. The three methods have similar relative data fitting error (about 10%). All the vector are scaled so that the largest entry has size 1. The solution provided by NuMF has the best performance for identifying the chemicals.

5.5 Chapter summary and perspectives

In this chapter, we introduced a new model of NMF, namely Nonnegative Unimodal Matrix Factorization (NuMF). We gave a characterization of the NU set, and proposed a brute-force heuristic to solve it, in which the heuristic is accelerated made by a dimension reduction step based on multi-grid method that is shown to preserves unimodality. We gave three preliminary results on the identifiability of NuMF under three special cases, and we present numerical experiments on synthetic and real datasets to illustrate the effectiveness of multi-grid and as well as giving empirical evidences to support the findings.

As the result on NuMF are preliminary, there are many open problems.

- **Restriction operators preserve NU and nonuniform grid** Currently we only show a special restriction operator preserves NU. Showing a class restriction operators preserve NU is still open. Furthermore, currently the restriction $\mathbf{R}$ performs a uniform grid on scaling down the problem. It will be interesting to investigate the use of nonuniform grid which is more adapted to the data structure on searching p.

- **General identifiability** Identifiability of NuMF in general remains open.

6 Tensor algebra and factorization

You can't learn too much linear algebra.

Benedict Gross

In this chapter we discuss tensor factorization and the solution approach to it.

Chapter organization We first give a review on the formalism of tensor for the purpose of this thesis in §6.1, where we give the notation and terminology of tensor algebra. Then, we focus on the tensor model named Canonical Polyadic Decomposition (CPD) in §6.2, where we showed that NMF and NTF problems are all constrained CPD problems. Finally, in §6.3 we discuss a general solution approach on solving CPD, namely the Block Coordinate Descent (BCD) algorithm framework.

6.1 A short review on the formalism of tensors

In this chapter, we review the formalism for tensors and present several concepts and notations in tensor decomposition. As stated in §1 , tensor problems arise as a generalization of matrix problems, the vast background of tensors makes it impossible to review all aspects. Hence here we only give the minimum necessary background on tensor for the purpose of this thesis. For a detail treatment of tensor, see for examples [62, 23, 97].

A N-way array or N-th order tensor $\mathcal{T}$ is a multidimensional array in the product $\mathbb{R}^{I_1} \times \cdots \times \mathbb{R}^{I_N}$ of the vector spaces $\mathbb{R}^{I_i}$ for $i \in [N]$, where $[N]$ denotes the integer set $\{1, 2, \dots, N\}$. A vector $\mathbf{x} \in \mathbb{R}_+^{I_1}$ is a 1st-order tensor, and a matrix $\mathbf{M} \in \mathbb{R}_+^{I_1 \times I_2}$ is a 2nd-order tensor. The data cubes shown in Fig.3.1 and Fig.3.4 are examples of 3rd-order tensors.

The Frobenius norm of a tensor $\mathcal{T}$ is defined as $\|\mathcal{T}\|_F = \sqrt{\sum_{j_1,\dots,j_N} \mathcal{T}^2_{j_1,\dots,j_N}}$, where the subscript notation $\mathcal{T}_{j_1\dots,j_N}$ denotes the $(j_1, \dots, j_N)-$th element of $\mathcal{T}$.

Products The tensor product $\bigotimes$ over N real vector spaces $\mathbb{R}^{I_1}, \dots, \mathbb{R}^{I_N}$ is defined as

$$\left[\bigotimes_{i=1}^{N} \mathbf{a}_p^{(i)}\right]_{j_1,\dots,j_N} := \prod_{i=1}^{N} \mathbf{a}_p^{(i)}(j_i), \quad \text{where } \mathbf{a}_p^{(i)} \in \mathbb{R}^{I_i} \text{ for all } i \in [N] \text{ and } p \in [r]. \tag{6.1}$$

The Kronecker product [14] of two matrices $\mathbf{A} \in \mathbb{R}^{I_1 \times J_1}$ and $\mathbf{B} \in \mathbb{R}^{I_2 \times J_2}$ is defined as

$$\mathbf{A} \boxtimes \mathbf{B} = \left[\begin{array}{c|c|c} A(1,1)\mathbf{B} & \dots & A(1,J_1)\mathbf{B} \\ \hline \vdots & \ddots & \vdots \\ \hline A(I_1,1)\mathbf{B} & \dots & A(I_1,J_1)\mathbf{B} \end{array}\right] \in \mathbb{R}^{I_1 I_2 \times J_1 J_2}. \tag{6.2}$$

Moreover, the Kronecker product of several matrices can be deduced from the Kronecker product (6.2) by associativity. The Khatri-Rao product $\mathbf{A} \odot \mathbf{B}$ is the column-wise Kronecker product. Setting $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_{J_1}]$ and $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_{J_1}]$,

$$\mathbf{A} \odot \mathbf{B} = \Big[\mathbf{a}_1 \boxtimes \mathbf{b}_1 \ , \ \cdots \ , \ \mathbf{a}_{J_1} \boxtimes \mathbf{b}_{J_1}\Big]. \tag{6.3}$$

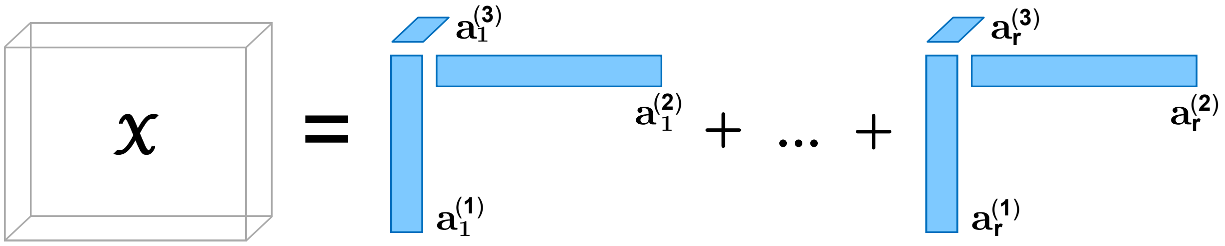


Fig. 6.1. Illustration of Equation (6.4a) with $N = 3$.

Compact notations of tensor decomposition There are several complementary notations to parameterize a low-rank tensor. In particular, grouping components $\mathbf{a}_p^{(i)}$ as columns of factor matrices $\mathbf{A}^{(i)} = [\mathbf{a}_1^{(i)}, \dots, \mathbf{a}_r^{(i)}]$, the following notations are equivalent:

$$\begin{aligned}
\mathcal{X} &= \sum_{p=1}^{r} \bigotimes_{i=1}^{N} \mathbf{a}_p^{(i)} && \text{(6.4a)}\\
&= [\![\mathbf{A}^{(1)}, \dots, \mathbf{A}^{(N)}]\!] && \text{(6.4b)}\\
&= \mathcal{I}_r \times_1 \mathbf{A}^{(1)} \times_2 \dots \times_N \mathbf{A}^{(N)} && \text{(6.4c)}\\
&:= \left(\bigotimes_{i=1}^{N}{}_a \mathbf{A}^{(i)} \right) \mathcal{I}_r, && \text{(6.4d)}
\end{aligned}$$

where $\mathcal{I}_r$ denotes the identity tensor, which the off-super-diagonal entries are 0s and the super-diagonal entries are 1s, $\times_p$ is the p-mode product (see [62]) and $\bigotimes_a$ is a tensor product of linear maps induced by the tensor product (6.1).

Fig. 6.1 illustrates an example of Equation (6.4a) for $N = 3$. Equation (6.4b) is the called *Kruskal notation*, equation (6.4c) makes use of the n-mode product, and equation (6.4d) uses the fact that linear applications on tensor spaces of finite dimensions also form a tensor space with tensor product

$$(\mathbf{A} \otimes_a \mathbf{B})\,(\mathbf{x} \otimes \mathbf{y}) := \mathbf{Ax} \otimes \mathbf{By}.$$

Since equation(6.4d) exhibits this tensor product structure, we stick with this formulation rather than the others.

Canonical Polyadic Decomposition and tensor rank The tensor decomposition listed above are called *Canonical Polyadic Decomposition* (CPD). Other kinds of tensor decomposition exist, for example, when $\mathcal{I}_r$ in equation (6.4d) is replaced by a general tensor $\mathcal{G}$, the model is called *Tucker Decomposition*:

$$\mathcal{T} = \left(\bigotimes_{i=1}^{N}{}_a \mathbf{A}^{(i)} \right) \mathcal{G}.$$

In this thesis, we mainly focus on CPD. However, we consider Tucker decomposition as a tool for compression for handling big data in §7.5.4.

In CPD, the rank of a tensor $\mathcal{X}$ is defined as the smallest r in equation (6.4a), i.e., $\mathcal{X}$ is the sum of r rank-1 tensors in the form $\bigotimes_{i=1}^{N} \mathbf{a}_p^{(i)}$. The problem of determining the rank of a tensor is called *Model Order Selection* which is a research topics on its own. Therefore, similar to the case on NMF described in §1.4.1 , in this thesis, we made the following (strong) assumption:

We assume the value of rank is given when solving an tensor decomposition problem.

Tensor unfoldings and useful formula To derive partial derivatives of a tensor function with respect to factors matrices, it is convenient to switch from a tensor formulation to the equivalent matrix description of the problem. More precisely, for tensor having a CPD structure, the following relationship holds:

$$\mathcal{X} = \left(\bigotimes_{i=1}^{N}{}_{a}\, \mathbf{A}^{(i)}\right)\mathcal{I}_r \qquad \equiv \qquad \forall i \in [N],\ \mathbf{X}_{[i]} = \mathbf{A}^{(i)}\left(\bigodot_{\substack{l\neq i\\ l=N}}^{1} \mathbf{A}^{(l)}\right)^{\top}, \tag{6.5}$$

where

$$\bigodot_{\substack{l\neq i\\ l=N}}^{1} \mathbf{A}^{(l)} = \mathbf{A}^{(N)}\mathbf{A}^{(N-1)}\ldots\mathbf{A}^{(i+1)}\mathbf{A}^{(i-1)}\ldots\mathbf{A}^{(1)},$$

and $\mathbf{X}_{[i]}$ denotes the unfolding of a rank-one tensor $\mathcal{X}$, and it is defined as

$$\mathbf{X}_{[i]} := \mathbf{a}^{(i)} \otimes \left(\boxtimes_{\substack{l\neq i\\ l=N}}^{1} \mathbf{a}^{(l)}\right) \in \mathbb{R}^{d}, \quad d = I_i \times \prod_{l\neq i} I_l. \tag{6.6}$$

Unfolding of a general tensor are obtained by linearity of the unfolding maps.

Finally, note that several non-equivalent definitions are used in the tensor signal processing community [62, 97, 22].

6.2 NMF and NTF problems as constrained CPD

In this section, we show that NMF and Nonnegative Tensor Factorization (NTF) problems are nonconvex optimization problems that are special cases of the following general CPD problem:

$$\underset{\substack{\mathbf{a}_p^{(i)}\\ i\in[N]\\ p\in[r]}}{\text{minimize}} \left\| \mathcal{T} - \sum_{p=1}^{r}\bigotimes_{i=1}^{N}\mathbf{a}_p^{(i)} \right\|_F^2 + \sum_{i,p} g_{i,p}(\mathbf{a}_p^{(i)}), \tag{6.7}$$

where $g_{i,p}$ are functions used to model regularizers and constraints. For example, for nonnegativity constraint, the function $g_{i,p}$ can be represented by the indicator function of the nonnegative orthant

$$i_{+}(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \geq 0 \\ \infty & \text{else} \end{cases}.$$

For $g = i_{+}(\cdot)$, if $N = 2$ problem (6.7) reduces to NMF problem (1.1), where

- $m = I_1$, $n = I_2$ and the tensor $\mathcal{T}$ is the matrix $\mathbf{M}$,
- the components $\mathbf{a}_1^{(1)}, \mathbf{a}_2^{(1)}, \ldots, \mathbf{a}_r^{(1)}$ are collectively grouped as the matrix $\mathbf{W}$,
- the components $\mathbf{a}_1^{(2)}, \mathbf{a}_2^{(2)}, \ldots, \mathbf{a}_r^{(2)}$ are collectively grouped as the matrix $\mathbf{H}$,
- the g functions correspond to the nonnegative constraints $\mathbf{W} \geq 0$ and $\mathbf{H} \geq 0$.

Note that problem (6.7) also includes the NMF models presented in §2 and §5 as special cases.

Recall in §1.4.3 we mentioned that NMF problem (1.1) can be solved by the two-block coordinate descent (BCD) scheme (Algorithm 1). Similarly, one approach to solve problem (6.7) is to use the same BCD scheme: we solve the subproblem with respect to one block of variables at a time while fixing the other variables. We will discuss more about BCD schemes in §6.3 .

6.2.1 The unconstrained CPD problem: an ill-posed problem

When there is no constraint in problem (6.7), we obtained the CPD problem, which is a difficult problem to solve, due to the following reasons.

- *Non-existence of solution and CP-degeneracy.* For CPD problem, a solution might not even exist, as the set of rank r tensors is not closed as soon as $r > 1$ and $p > 2$ [27]. When an optimal CPD solution does not exist, then any sequence of CPD factors that monotonically decreases the cost function value will exhibit the so-called CP-degeneracy [63]. The term CP-degeneracy refers to that the estimates of each block $\mathbf{A}^{(j)} = [\mathbf{a}_1^{(j)}, \dots, \mathbf{a}_r^{(j)}]$, $(j \in [p])$ have some columns growing to infinity while canceling each other [63, 27, 85, 92, 23].

- *Swamps.* Swamp is a word coined by R. Harshman and others that describes the behavior of the decrease of the cost function value [92]. A swamp is when a group of vectors in $\{\mathbf{a}_p^{(i)}\}$ are nearly linearly dependent, which has been observed to result in slow convergence of algorithms [48]. Note that the existence of swamps when solving CPD problem is well documented [85, 92, 23, 48], and it has been observed that CP-degeneracies cause swamps, however, the mechanism of what causes the swamp, is still not fully understood [48]. We refer to the papers [85, 92, 23, 48] for more discussions on this issue.

In this thesis, we use the term “swamps” in a more general fashion, we simply refer to the periods of slow convergence as swamps. Many figures in §7.6 show the existence of swamps.

Hence, in general CPD is a hard problem. Note that the nonnegative CPD (i.e. NTF), is NP-hard because it is a generalization of NMF which is NP-hard (see §1.4.6). In general, computing CPD and NTF are challenging tasks.

Remark 6. *We give a minor remark on the non-existence of optimal solution. Note that it can pose problems in practice in minimizing the cost function, as mentioned above, but the non-existence of optimal solution can actually also be seen as an advantage: we may simply use it to tell whether a tensor is CPD-factorizable or not. Note that this is not possible for the matrix case.*

6.3 Block-Coordinate Descent algorithms

As discussed in the last section, the nonconvex problem (6.7) is in general NP-hard and no closed-form solution is known to solve it, therefore there has been a large amount of works dedicated on approximately solving NTF using various optimization heuristics. While it is impossible to review all the heuristics, we focus on a class of algorithms, namely the Block-Coordinate Descent (BCD). For a short review on other types of algorithms on solving problem (6.7), such as second order method and randomized methods, see [2, Section 2]. From now on, the functions g in problem (6.7) are indicator function of nonnegative orthant.

Cost function and gradient with respect to a block, and the MTTKRP First we give one important expression related to problem (6.7): the derivative with respect to one block. On problem (6.7), using the equivalence between expressions (6.4a) and (6.4d), and expressing nonnegativity constraint explicitly, we can express the Nonnegative Tensor Factorization (NTF) problem using a block-form expression for the cost function as follows

$$\min_{\mathbf{A}^{(i)}\geq 0, i\in[N]} \; F(\mathbf{A}^{(1)},\dots,\mathbf{A}^{(N)}) := \frac{1}{2}\left\|\mathcal{T} - \left(\bigotimes_{i=1}^{N}{}_{a}\, \mathbf{A}^{(i)}\right)\mathcal{I}_r\right\|_F^2. \tag{6.8}$$

To derive the gradient of F with respect to the single block $\mathbf{A}^{(i)}$, we make use of the Khatri-Rao product (6.5). First, let

$$\mathbf{B}^{(i)} := \mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(i+1)} \odot \mathbf{A}^{(i-1)} \odot \cdots \odot \mathbf{A}^{(1)}, \tag{6.9}$$

then the gradient of F w.r.t. the single block $\mathbf{A}^{(i)}$ is given by

$$\nabla_{\mathbf{A}^{(i)}} F = \Big(\mathbf{A}^{(i)}(\mathbf{B}^{(i)})^\top - \mathcal{T}_{[i]}\Big)\mathbf{B}^{(i)}. \tag{6.10}$$

This expression contains the bottleneck operation in many tensor algorithms: the Matricized Tensor Times Khatri-Rao Product (MTTKRP). As we can see that, to compute $\nabla_{\mathbf{A}^{(i)}} F$, we need to multiply the matricized tensor $\mathcal{T}_{[i]}$ with the Khatri-Rao product $\mathbf{B}^{(i)}$. Such operation is expensive but unavoidable. In fact, the number of MTTKRP operations in an algorithm can often be used as a criterion to judge how computationally expensive it is to run that algorithm.

BCD methods BCD has become a standard and efficient scheme for solving NTF, mainly because

1. it essentially has cheap computation cost in each block update (BCD fixes all blocks except for one),
2. BCD can make use of recent developments in convex constrained optimization to efficiently solve NTF with respect to each block,
3. under some suitable assumptions, many 1st-order BCDs and their accelerated versions have convergence guarantee in the context of general block-separable *nonconvex composite optimization* problem that subsumes NTF as a special case [109, 54].

Below, we review several BCD methods for solving NTF. In §6.3.1 , we review AO-AS [61] , AO-ADMM [55] , AO-Nesterov [110] and (AO)-A-HALS [46] . In §6.3.2 , we review APG [109] and iBPG [54] .

6.3.1 Alternating optimization framework

When solving NTF using BCDs, the blocks of variables that are alternatively updated must be chosen. The cost function F in NTF problem (6.8) is a quadratic function with respect to each matrix $\mathbf{A}^{(i)}$, so the optimization subproblem

$$\min_{\mathbf{A}^{(i)}\geq 0} \; F(\mathbf{A}^{(1)},\dots,\mathbf{A}^{(N)})$$

is a linearly constrained QP known as the Nonnegative Least Squares (NNLS). In particular, it is strictly convex if $\mathbf{B}^{(p)}$ (see definition (6.9)) is full column-rank. Therefore, it is quite natural to consider $\mathbf{A}^{(i)}$ as the blocks in a BCD. The Alternating optimization (AO) framework, which is a standard procedure to solve NTF, alternatively (exactly/inexactly) solves subproblem for each block. We describe the AO framework in Algorithm 6 . Note that the objective function of AO methods decreases after each block update. Depending on how the matrix-form NNLS problem (6.11) is solved, various implementations of AO algorithms can be obtained. Some of them are very efficient for solving NTF, they are surveyed below.

Algorithm 6 Alternating optimization framework

```
Input: a nonnegative $N$-way tensor
Output: nonnegative factors $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \dots, \mathbf{A}^{(N)}$.
Initialization: $(\mathbf{A}_0^{(1)}, \dots, \mathbf{A}_0^{(N)})$.
for $k = 1, \dots$ until some criteria is satisfied do
    for $i = 1, \dots, N$ do
        Update $\mathbf{A}_k^{(i)}$ as an exact/inexact solution of
            $$\min_{\mathbf{A}^{(i)} \geq 0} F\Big(\mathbf{A}_k^{(1)}, \dots, \mathbf{A}_k^{(i-1)}, \mathbf{A}^{(i)}, \mathbf{A}_{k-1}^{(i+1)}, \dots, \mathbf{A}_{k-1}^{(N)}\Big). \qquad (6.11)$$
    ($\mathbf{A}_{k-1}^{(i)}$ can be used as the initial point for the algorithm used to solve (6.11).)
    end for
end for
```

AO-AS – solving NNLS with Active Set When $\mathbf{A}_k^{(i)}$ is updated by an exact solution of the NNLS subproblem (6.11), we obtain an alternating NNLS algorithm, usually referred to as ANLS in the literature. To solve exactly the NNLS subproblem (6.11), active set (AS) methods are usually rather effective and popular [61]. We will refer to AO-AS as the ANLS algorithm where the NNLS subproblems are solved with AS.

AO-ADMM – solving NNLS with ADMM Designed to tackle a wide range of constrained tensor decomposition problems and various loss functions, AO-ADMM [55] applied to NTF boils down to using several steps of a primal-dual algorithm, the Alternating Direction Method of Multipliers (ADMM), to solve the cascaded NNLS problems. Therefore, AO-ADMM for NTF problem (6.7) is a variant of the AO framework that solves NNLS subproblem (6.11) inexactly.

AO-Nesterov When Nesterov's accelerated gradient method is applied to solve the NNLS subproblem (6.11), we obtain AO-Nesterov [110, 51] . An example is the AccProjG presented in §2.2.1 on solving NMF subproblem on block $\mathbf{H}$.

A-HALS While we introduced HALS in §1.4.3, here we re-introduce it again in the language of NTF. The *Hierarchical Alternating Least Square* (HALS) algorithm was introduced for solving NMF problem, and has been widely used for solving NMF as it performs extremely well in practice [44, 21].

HALS cyclically updates each column of the factor matrix $\mathbf{A}^{(i)}$ by solving an NNLS problem with respect to that column while fixing the others. The optimal solution of this NNLS subproblem can be written in closed form. A-HALS, which is short for accelerated HALS, was proposed in [46] to accelerate HALS. A-HALS repeats updating each factor matrix several times before updating the other ones. Hence A-HALS can be considered as a variant of the AO framework where each NNLS is inexactly solved itself by a BCD with closed-form updates. Let us briefly describe A-HALS for solving NTF. The NNLS subproblem (6.11) of A-HALS is inexactly solved by repeating cyclically updating the columns of $\mathbf{A}_{k-1}^{(i)}$. In particular, based on the relationship (6.5)

$$\mathbf{M} = \mathcal{T}_{[i]}, \quad \mathbf{W} = \mathbf{A}_{k-1}^{(i)}, \quad \mathbf{H} = \left(\bigodot_{\substack{l \neq i \\ l=N}}^{1} \mathbf{A}^{(l)} \right)^{\top} = \left(\mathbf{A}_{k-1}^{(N)} \odot \dots \mathbf{A}_{k-1}^{(i+1)} \odot \mathbf{A}_{k}^{(i-1)} \dots \odot \mathbf{A}_{k}^{(1)} \right)^{\top},$$

then the j-th column of $\mathbf{A}_{k-1}^{(i)}$, which is now denoted as $\mathbf{W}(:,j)$ is updated by

$$\mathbf{W}(:,j) = \frac{\left[\mathbf{M}\mathbf{H}(j,:)^{\top} - \sum_{l \neq j} \mathbf{W}(:,l)\mathbf{H}(l,:)\mathbf{H}(j,:)^{\top}\right]_{+}}{\|\mathbf{H}(j,:)\|_2^2}.$$

It is worth noting that, under some mild conditions, A-HALS for NTF has sub-sequential convergence guarantee (i.e., every limit point is a stationary point of the objective function); see [93, Section 7].

6.3.2 Block proximal gradient type methods

The NNLS subproblem (6.11) does not have a closed-form solution. From the gradient expression (6.10), the function F restricted to $\mathbf{A}^{(i)}$, is a $L^{(i)}$-smooth function, that is, the gradient $\nabla_{\mathbf{A}^{(i)}} F$ is Lipschitz continuous with the constant $L^{(i)} = \|(\mathbf{B}^{(i)})^{\top}\mathbf{B}^{(i)}\|$, where $\mathbf{B}^{(i)}$ is defined in definition (6.9). This property can be employed to replace the objective function in the NNLS subproblem (6.11) by its quadratic majorization function, that leads to a new minimization problem which has a closed-form solution. This minimization-majorization approach, in the literature of block-separable composite optimization problem with the block-wise L-smooth property, is known as proximal gradient block coordinate descent method [54]. Considering the NTF problem, the closed-form solution of minimizing the majorization function is a projected gradient step. Applying Nesterov-type acceleration for the proximal gradient step improves the convergence of the BCD algorithm. Below we review the two recent accelerated proximal gradient BCD methods that were proposed for solving the general composite optimization problem.

APG – An Alternating Proximal Gradient method for solving NTF APG was proposed in [109]; see the Appendix of [3] and [109, Section 3.2] for the algorithm pseudocode. APG *cyclically* updates each block $\mathbf{A}^{(i)}$ by calculating an extrapolation point $\hat{\mathbf{A}}_{k-1}^{(i)} = \mathbf{A}_{k-1}^{(i)} + w_{k-1}^{(i)}\left(\mathbf{A}_{k-1}^{(i)} - \mathbf{A}_{k-2}^{(i)}\right)$ (here $w_{k-1}^{(i)}$ is some extrapolation parameter) and embedding this point in a projected gradient step

$$\mathbf{A}_{k}^{(i)} = \left[\hat{\mathbf{A}}_{k-1}^{(i)} - \frac{1}{L_{k-1}^{(i)}} \left(\hat{\mathbf{A}}_{k-1}^{(i)} (\mathbf{B}_{k-1}^{(i)})^{\top} - \mathcal{T}_{[i]} \right) \mathbf{B}_{k-1}^{(i)} \right]_{+}.$$

After all blocks are updated, APG needs a restarting step, that is, if the objective function has increased then the projected gradient step would be re-done by using the previous values of all blocks instead of using the extrapolation points.

iBPG – An inertial Block Proximal Gradient Method Proposed in [54], iBPG computes two different extrapolation points $\hat{\mathbf{A}}_{k-1}^{(i,1)}$ and $\hat{\mathbf{A}}_{k-1}^{(i,2)}$: one is for evaluating the gradient and the other one for adding inertial force. iBPG updates one matrix factor using a projected gradient step

$$\mathbf{A}_k^{(i)} = \left[\hat{\mathbf{A}}_{k-1}^{(i,2)} - \frac{1}{L_{k-1}^{(i)}} \Big(\hat{\mathbf{A}}_{k-1}^{(i,1)} (\mathbf{B}_{k-1}^{(i)})^\top - \mathcal{T}_{[i]} \Big) \mathbf{B}_{k-1}^{(i)} \right]_+ ,$$

see the Appendix of [3] for the algorithm pseudocode. Furthermore, similarly to A-HALS, iBPG allows updating each matrix factor some times before updating another one – this feature would help save some computational costs since some common expressions can be re-used when repeating updating the same block. iBPG does not require a restarting step which make it suitable for solving large-scale NTF problems where evaluating the objective functions is costly.

Difference between APG, iBPG and AO-Nesterov APG, iBPG and AO-Nesterov are all gradient-based methods, however they are different in the following aspects:

- In AO-Nesterov, each subproblem is solved using a Nesterov-type accelerated gradient method, i.e., we perform gradient update on the same block several times, before switching to other block. This is different from APG and iBPG that they update on one block using a gradient step, then move to another block immediately. Furthermore, under the theory of iBPG, the gradient update on the same block in iBPG method can be repeated multiple times, and this is not covered nor considered in the theory in APG.
- The ways of computing extrapolation coefficient between AO-Nesterov, APG and iBPG are slightly different.

Chapter summary

We discussed the formalism of tensors, the CPD formulation, and reviewed some algorithms that solve CPD problems under the BCD framework.

7 Heuristic extrapolation with restarts

> It is possible to invent a single machine which can be used to compute any computable sequence.
>
> *Alan Turing*

In this chapter we introduce an acceleration framework, namely Heuristic extrapolation with restarts (HER), for accelerating the algorithms for solving NMF and NTF. HER was first proposed to solve NMF problem in [5], and the HER framework was only applied on A-HALS and Active Set method in [5] (also see §1.4.3), then extended to tensor problems in [1, 3, 2]. HER was inspired from the method of parallel tangents which is closely related to the conjugate gradient method [80, p. 293], and the accelerated gradient methods by [89], which has now become the standard technology in various optimization scenarios [10].

Chapter organization This chapter is organized as follows. First we give the idea of acceleration through extrapolation in §7.1 . Then, in §7.2, we present the original form of HER for NMF algorithms. Next, we present HER framework for general tensor problems in §7.3 . After that, we present the numerical results of HER compared with other algorithms on NMF, NTF and CPD problems in §7.4 , §7.5,§7.6 , respectively. Finally, we summarize this chapter in §7.7 , where we also present some open problems. As this chapter is a long, we give a diagram to illustrate the relationships between the sections of this chapter, see Fig.7.1 .

Highlights of contributions The whole chapter contains the contribution. We introduce the HER framework, presented in the original form for NMF in §7.2 , and in the form for NTF and CPD in §7.3 . In the remaining parts of the chapter, we present the comparative study on the effectiveness of HER.

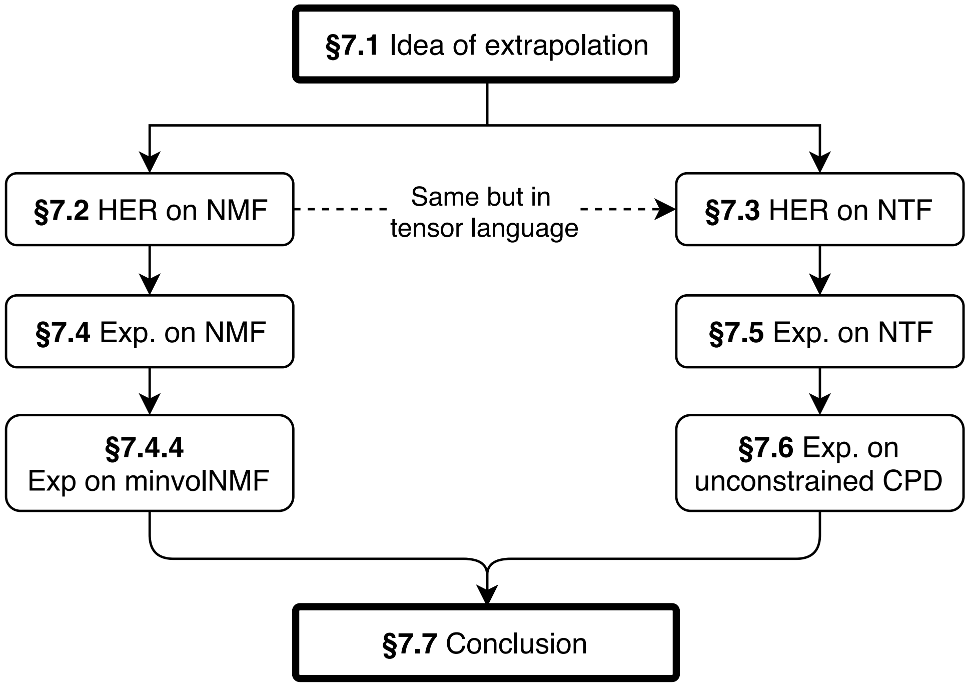


Fig. 7.1. Structure of chapter 7.

7.1 The idea of acceleration through extrapolation

In this section we lay-down the idea of extrapolation for the purpose of accelerating algorithms.

Assume we have an optimization scheme that computes the next iterate only based on the previous iterate[1] (e.g., gradient descent or a coordinate descent), i.e., it updates the kth iterate $\mathbf{x}_k$ as $\mathbf{x}_{k+1} = \texttt{update}(\mathbf{x}_k)$, for some function `update` that depends on the objective function and the feasible set.

For 1st-order methods, if the landscape of the objective function is elliptic, the updates will have a zigzagging behavior. In particular, gradient descent with exact line search leads to orthogonal search directions [80] while search direction of (block) coordinate descent methods are orthogonal by construction. Fig.7.2 below illustrate the zigzagging behavior of the gradient descent update in solving a simple quadratic function.

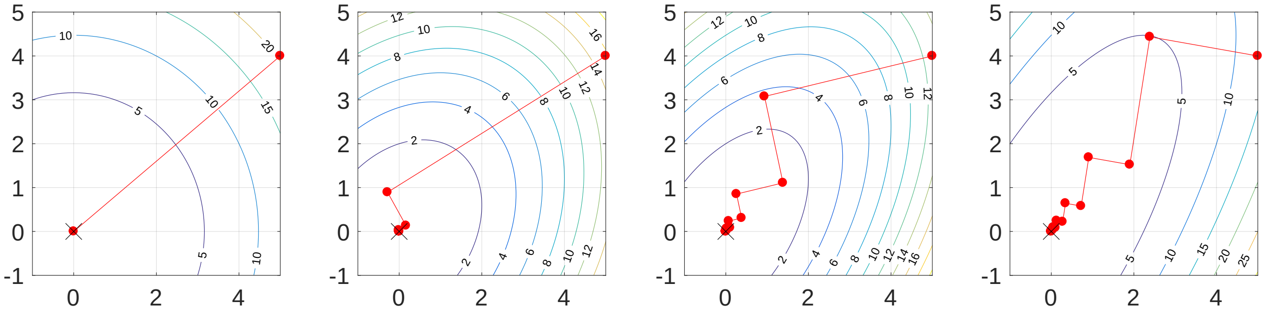


Fig. 7.2. Illustration of the zigzagging behavior of the gradient descent update in solving a simple quadratic function. Here the problem is to minimize the quadratic form $\frac{1}{2}\langle \mathbf{x}, \mathbf{Q}\mathbf{x}\rangle$ with four different $\mathbf{Q}$, using gradient descent with the same starting point. From left right, the condition number of $\mathbf{Q}$ is increasing, and the landscape of the objective function (represented by the labeled contour curves) is changing from circular to more and more elliptic. We can see that, when the landscape of the objective function is elliptic, gradient updates will have a strong zigzagging behavior, and the zigzagging behavior slows down the convergence of the sequence.

The use of extrapolation is to reduces the zigzagging for the purpose of acceleration. The idea is to define a second sequence of iterates, namely, $\mathbf{y}_k$ with $\mathbf{y}_0 = \mathbf{x}_0$, and modify the update scheme as

$$\mathbf{x}_{k+1} = \texttt{update}(\mathbf{y}_k), \quad \mathbf{y}_{k+1} = \mathbf{x}_{k+1} + \beta_k(\mathbf{x}_{k+1} - \mathbf{x}_k),$$

for some $\beta_k \geq 0$. Note that there are other possibilities for choosing $\mathbf{y}_{k+1}$ based on a linear combinations of previous iterates, and note that the iterates $\mathbf{y}_k$'s do not necessarily belong to the feasible set.

Pictorial illustration Fig. 7.3 illustrates the extrapolation scheme, and allows us to get some intuition: the direction $(\mathbf{x}_{k+1} - \mathbf{x}_k)$ will be in-between zigzagging directions obtained with the original update applied to $\mathbf{y}_k$'s and will allow to accelerate convergence. For example, we observe on Fig. 7.3 that the direction $\mathbf{x}_{k+2} - \mathbf{x}_{k+1}$ is between the directions $\mathbf{x}_{k+1} - \mathbf{y}_k$ and $\mathbf{x}_{k+2} - \mathbf{y}_{k+1}$.

Extrapolation in convex optimization In the case of gradient descent and convex optimization, the β_k's can be chosen a priori for obtaining the theoretical acceleration, and the above scheme allows to

[1] Although this assumption is not strictly necessary, it makes more sense otherwise there might be a counter effect if the update already takes into account the previous iterates.

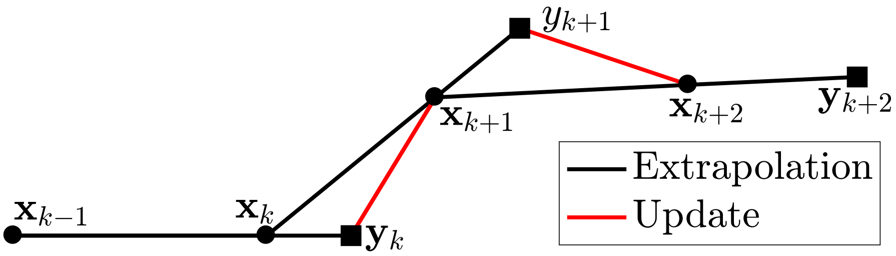


Fig. 7.3. Illustration of the idea of extrapolation to accelerate optimization schemes.

accelerate convergence of the function values from $\mathcal{O}(1/k)$ to $\mathcal{O}(1/k^2)$, and from linear convergence with rate $1-\mu/L$ to rate $1-\sqrt{\mu/L}$ for strongly convex function with parameter μ and whose gradient has Lipschitz constant L [89]. In nonsmooth convex optimization, similar results can be obtained by using the proximal operator and the accelerated proximal gradient methods [10].

Extrapolation in nonconvex optimization In the nonconvex case, the scheme has also been used for BCD, and most works focus on the case where the blocks of variables are updated using a gradient or proximal gradient step; e.g., [109, 54]. However, from a practical point of view, the acceleration depends on the choice of the β_k's which is highly non-trivial. In fact, even in the convex case, the best choice for the β_k's is difficult in practice; see, e.g., [91] for a discussion about this issue. As far as we know, extrapolation has not been used in combination with exact BCD methods. Both works in [109, 54] used extrapolation in the context of an inexact BCD method where the blocks of variables are updated using a projected gradient method. Their approach is different from HER as it can be used with exact BCD. Note that we will compare HER with [109, 54] applied to NMF, NTF and CPD problems in the subsequent sections in this chapter.

In the method of parallel tangents, the steps β_k are computed using line-search [80, p. 293]. This allows the acceleration scheme to be at least as good as the initial scheme. However, as we will see, this is not a good strategy in our case as the optimal β_k's will be close to zero (because we use coordinate descent). In any case, the choice of the β_k's is highly non-trivial and, as we will see, the acceleration depends on the choice of these parameters. Note that choosing $\beta_k = 0$ for all k gives back the original algorithm (no extrapolation), and β_k close to one is a very aggressive strategy.

7.2 Extrapolation for NMF algorithms

In this section, we present the original form of HER used to accelerate algorithms that solves NMF problem (1.1). This section serves two purposes, (i) it let us understand the intuition behind the design of the HER framework, and (ii) it acts as an introductory phase for the HER algorithm for solving NTF and CPD.

The (three) algorithm(s) We adapt extrapolation to the two-BCD strategies of NMF algorithms described in Algorithm 1. Algorithm 7 describes the proposed extrapolation scheme applied to NMF problem (1.1). Depending on the choice of the parameter $hp \in \{1, 2, 3\}$, (h stands for $\mathbf{H}$, p for projection), Algorithm 7 corresponds to three different variants of the proposed extrapolation. This is described below through two important questions. For ease of understanding, Fig.7.4 shows the flow of line 4 to line 16 of Algorithm 7 for the parameter $hp \in \{1, 2, 3\}$.

Algorithm 7 Acceleration of Algorithm 1 using extrapolation (The original form of HER)

```
Input: A matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$, parameters $hp \in \{1,2,3\}$ (extrapolation/projection option of $\mathbf{H}$). Parameters $1 < \bar{\gamma} < \gamma < \eta$, $\beta_1 \in (0,1)$.
Output: An approximate solution $(\mathbf{W},\mathbf{H})$ to NMF (1.1).
Initialization: $\mathbf{W} \in \mathbb{R}^{m\times r}_{+}$, $\mathbf{H} \in \mathbb{R}^{m\times r}_{+}$, $\hat{\mathbf{W}} = \mathbf{W}$; $\hat{\mathbf{H}} = \mathbf{H}$; $\hat{e}(0) = \|\mathbf{M} - \mathbf{W}\mathbf{H}\|_F$. Set $\bar{\beta} = 1$.
for $k = 1, \dots$ until some criteria is satisfied do
    Compute $\mathbf{H}_{\text{new}}$ using a NNLS algorithm to solve $\underset{\mathbf{H}\geq 0}{\text{argmin}}\ ||\mathbf{M} - \hat{\mathbf{W}}\mathbf{H}||_F^2$ using $\hat{\mathbf{H}}$ as the initial iterate.
    if $hp \geq 2$ then
        $\hat{\mathbf{H}} = \mathbf{H}_{\text{new}} + \beta_k(\mathbf{H}_{\text{new}} - \mathbf{H})$.        % Extrapolation
    end if
    if $hp = 3$ then
        $\hat{\mathbf{H}} = [\hat{\mathbf{H}}]_+$.        % Projection
    end if
    Compute $\mathbf{W}_{\text{new}}$ using a NNLS algorithm to solve $\underset{\mathbf{W}\geq 0}{\text{argmin}}\ ||\mathbf{M} - \mathbf{W}\hat{\mathbf{H}}||_F^2$ with $\mathbf{W} \geq 0$ using $\hat{\mathbf{W}}$ as the initial iterate.
    $\hat{\mathbf{W}} = \mathbf{W}_{\text{new}} + \beta_k(\mathbf{W}_{\text{new}} - \mathbf{W})$.        % Extrapolation
    if $hp = 1$ then
        $\hat{\mathbf{H}} = \mathbf{H}_{\text{new}} + \beta_k(\mathbf{H}_{\text{new}} - \mathbf{H})$.        % Extrapolation
    end if
    Compute error: $\hat{e}(k) = ||\mathbf{M} - \mathbf{W}_{\text{new}}\hat{\mathbf{H}}||_F$.        % See Remark 7.
    if $\hat{e}(k) > \hat{e}(k-1)$ then
        $\hat{\mathbf{H}} = \mathbf{H}$; $\hat{\mathbf{W}} = \mathbf{W}$.        % Restart
        $\beta_{k+1} = \beta_k/\eta$.        % Decrease the value of $\beta$.
        $\bar{\beta} = \beta_{k-1}$.
    else
        $\mathbf{H} = \mathbf{H}_{\text{new}}$; $\mathbf{W} = \mathbf{W}_{\text{new}}$.
        $\beta_{k+1} = \min(\bar{\beta}, \gamma\beta_k)$.        % Increase the value of $\beta$
        $\bar{\beta} = \min\left(1, \bar{\gamma}\bar{\beta}\right)$.
    end if
end for
```

When should we perform the extrapolation? In NMF (and in general for BCD methods), it makes sense to perform the extrapolation scheme after the update of each block of variables so that when we update the next block of variables, the algorithm takes into account the already extrapolated variables; see, e.g., [35]. However, as we will see in the experiments, this does not necessarily performs best in all cases. This is the first reason why we added a parameter $hp \in \{1,2,3\}$: For $hp = 1$, $\mathbf{H}$ is extrapolated after the update of $\mathbf{W}$, otherwise it is extrapolated directly after it has been updated. Note that in the former case, the extrapolated matrix $\hat{\mathbf{H}}$ is only used as a warm start for the next NNLS update of $\mathbf{H}$. For ANLS (algorithms using active-set method to solve the NNLS, see §1.4.3), it will therefore not play a crucial role as ANLS solves the NNLS subproblem exactly.

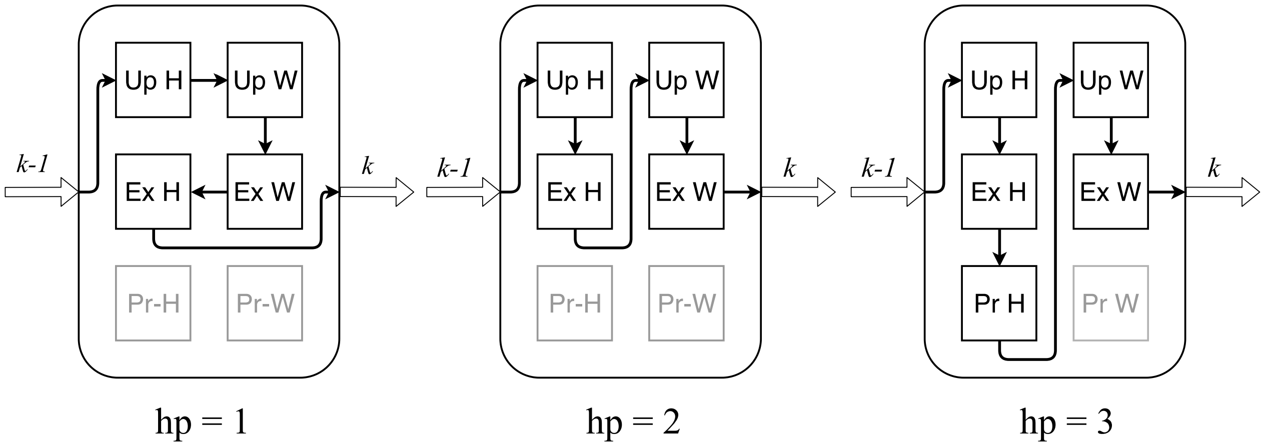


Fig. 7.4. The flow of line 4 to line 16 of Algorithm 7 for the parameter $hp \in \{1,2,3\}$. In the figure, k is the iteration number, “Up" refers to “update”, “Ex" refers to “extrapolate” and “Pr” refers to “project”.

Can we guarantee convergence? Currently there is no theory to guaranteed Algorithm 7 converges to stationary points. In Algorithm 7, because $\hat{\mathbf{W}}$ and $\hat{\mathbf{H}}$ are not necessarily nonnegative, the objective function is not guaranteed to decrease at each step. In fact, step 18 of Algorithm 7 only checks the decrease of $||\mathbf{M} - \mathbf{W}_{\text{new}}\hat{\mathbf{H}}||_F$ where $(\mathbf{W}_{\text{new}}, \hat{\mathbf{H}})$ is not necessarily feasible for $hp \geq 2$. The reason for computing $||\mathbf{M} - \mathbf{W}_{\text{new}}\hat{\mathbf{H}}||_F$ and not $||\mathbf{M} - \mathbf{W}_{\text{new}}\mathbf{H}_{\text{new}}||_F$ is threefold:

- $\mathbf{W}_{\text{new}}$ was updated according to $\hat{\mathbf{H}}$ (line 12 in Algorithm 7).

- It gives the algorithm some degrees of freedom to possibly increase the objective function in the hope to be able to decrease it significantly later on: in fact, we observed in experiments that this choice allows a faster convergence than when restarting the algorithm based on the error $||\mathbf{M} - \mathbf{W}_{\text{new}}\mathbf{H}_{\text{new}}||_F$,

- It is computationally cheaper because computing $||\mathbf{M} - \mathbf{W}_{\text{new}}\mathbf{H}_{\text{new}}||_F$ would require $\mathcal{O}(mnr)$ operations instead of $\mathcal{O}(mr^2)$ (see Remark 7). This point becomes much more important when we generalize this algorithm framework from NMF to the tensor cases in §7.3.

To guarantee the objective function to decrease, one way is to require $\hat{\mathbf{H}}$ to be nonnegative by projecting it to the nonnegative orthant: this variant corresponds to $hp = 3$. In that case, the solution $(\mathbf{W}_{\text{new}}, \hat{\mathbf{H}})$ is a feasible solution for which the objective function is guaranteed to decrease at least every second step. In fact, when the error increases, Algorithm 7 re-initializes the extrapolation sequence $(\hat{\mathbf{W}}, \hat{\mathbf{H}})$ using $(\mathbf{W}, \mathbf{H})$ (step 19 of Algorithm 7) and the next step is a standard NNLS update. So, since the objective function is bounded below, there exists a converging subsequence of the iterates. Proving convergence to a stationary points is an open problem, and an important direction for further research. We believe it would be interesting to investigate the convergence of the extrapolation scheme applied on BCD in the nonconvex case.

To summarize, using the extrapolation of $\mathbf{H}$ after the update of $\mathbf{W}$ ($hp = 1$) or using the projection of $\hat{\mathbf{H}}$ onto the feasible set ($hp = 3$) is more conservative but guarantees the objective function to decrease (at least every second step). As we will see in the experiments, these two variants perform in general better than with $hp = 2$.

Remark 7 (Computation of the error)**.** *To compute the error $\hat{e} = ||\mathbf{M} - \mathbf{W}_{new}\hat{\mathbf{H}}||_F^2$ in step 17 of Algorithm 7, it is important to take advantage of previous computations and not compute $\mathbf{W}_{new}\hat{\mathbf{H}}$ explicitly (which would be impractical for large sparse matrices). For simplicity, let us denote $\mathbf{W} = \mathbf{W}_{new}$ and $\mathbf{H} = \hat{\mathbf{H}}$, then*

$$||\mathbf{M} - \mathbf{W}\mathbf{H}||_F^2 = ||\mathbf{M}||_F^2 - 2\langle \mathbf{W}, \mathbf{M}\mathbf{H}^\top \rangle + \langle \mathbf{W}^\top \mathbf{W}, \mathbf{H}\mathbf{H}^\top \rangle.$$

The term $||\mathbf{M}||_F^2$ can be computed once, the term $\langle \mathbf{W}, \mathbf{M}\mathbf{H}^\top \rangle$ can be computed in $\mathcal{O}(mr)$ operations since $\mathbf{M}\mathbf{H}^\top$ is computed within the NNLS update of $\mathbf{W}$, and the term $\langle \mathbf{W}^\top \mathbf{W}, \mathbf{H}\mathbf{H}^\top \rangle$ requires $\mathcal{O}(mr^2)$ since $\mathbf{H}\mathbf{H}^\top$ is also computed within the NNLS update of W. In fact, all algorithms for NNLS we know need to compute $\mathbf{M}\mathbf{H}^\top$ and $\mathbf{H}\mathbf{H}^\top$ when solving for $\mathbf{W}$, because the gradient of $||\mathbf{M} - \mathbf{W}\mathbf{H}||_F^2$ with respect to $\mathbf{W}$ is $2(\mathbf{W}\mathbf{H}\mathbf{H}^\top - \mathbf{M}\mathbf{H}^T)$.

Remark 8 (Other NMF variants)**.** *Although here we focus on the standard NMF*(1.1)*, the acceleration scheme described in Algorithm 7 can be directly applied to any NMF model, such as the minvol NMF presented in §2.1, and the NuMF in §5.*

7.2.1 Choice of the extrapolation parameters β_k's

Here, we propose a strategy to choose the β_k's. First, we explain why it does not work well to use line search. Let us focus on the update of $\mathbf{W}$ (a similar argument holds for $\mathbf{H}$), then

$$\hat{\mathbf{W}} = \hat{\mathbf{W}}(\beta) = \mathbf{W}_{\text{new}} + \beta(\mathbf{W}_{\text{new}} - \mathbf{W}),$$

where $\mathbf{W}_{\text{new}}$ is an approximate solution of $\min_{\mathbf{W}\geq 0} ||\mathbf{M} - \mathbf{W}\hat{\mathbf{H}}||_F$ (in the case of ANLS, it is an optimal solution). Note that if $\beta = -1$, it gives $\hat{\mathbf{W}} = \mathbf{W}$ and if $\beta = 0$, it gives $\hat{\mathbf{W}} = \mathbf{W}_{\text{new}}$.

The optimal β can be computed in closed-form as follows

$$\beta^* = \underset{\beta}{\operatorname{argmin}} \; ||\mathbf{M} - \hat{\mathbf{W}}(\beta)\hat{\mathbf{H}}||_F^2 = \frac{\langle \mathbf{M} - \mathbf{W}_{\text{new}}\hat{\mathbf{H}}, (\mathbf{W}_{\text{new}} - \mathbf{W})\hat{\mathbf{H}} \rangle}{||(\mathbf{W}_{\text{new}} - \mathbf{W})\hat{\mathbf{H}}||_F^2}.$$

We see the followings.

- **Computing β^* is expensive for large matrices**.
- **The β^* are mostly zero**. Empirically, for most steps of Algorithm 7 , we observed that β^* is close to zero[2], especially when the algorithm has performed several iterations and reached the neighborhood of a stationary point. The reason is that $\mathbf{W}_{\text{new}}$ was optimized to minimize the objective function.

To summarize, the line search approach is empirically not providing much push forces on the iterates, while being practically expensive to compute.

Strategy for updating the β_k's In the following, we propose another strategy to tune β_k's. It will increase the objective function in most cases (i.e., $||\mathbf{M} - \hat{\mathbf{W}}(\beta)\hat{\mathbf{H}}||_F^2 > ||\mathbf{M} - \hat{\mathbf{W}}(0)\hat{\mathbf{H}}||_F^2$) but will allow a larger decrease of the objective function at the next step. Note that this is the reason why we check whether the error has decreased only after the update of $\mathbf{H}$ because otherwise the acceleration would not be possible (only a small β would be allowed in that case).

[2] β^* is not always close to zero–even when using ANLS–as $\hat{\mathbf{W}}$ is not necessarily nonnegative.

Note that, as we are applying the extrapolation scheme to a nonconvex problem using BCD, there is, as far as we know, no a priori theoretically sound choice for the β_k's. For this reason, we consider a simple scheme described by lines 20,21,24,25 in Algorithm 7 . It works as follows. Assume there exists a hidden optimal value for the β_k's, like in the strongly convex case where β_k should ideally be equal to $\frac{1-\sqrt{\mu/L}}{1+\sqrt{\mu/L}}$ [89], where μ is the strong convexity parameter of the objective function and L is the Lipschitz constant of its gradient. It starts with an initial value of $\beta_0 \in [0,1]$, and an upper bound $\bar{\beta} = 1$. Note that initially, the algorithm can take a rather large steps since usually the initialization $(\mathbf{W}, \mathbf{H})$ is far from being locally optimal, hence it makes sense for β_0 to be chosen in $[0,1]$. As long as the error decreases, the scheme increases the value of β_{k+1} by a factor γ taking into account the upper bound (line 24). It also increases the upper bound by a factor $\bar{\gamma} < \gamma$ if it is smaller than 1 (line 25). The usefulness of $\bar{\beta}$ is to keep in memory the last value of β_k that allowed decrease of the objective function which is used as an upper bound for β_k. However, because the landscape of the objective function may change, $\bar{\beta}$ is slightly increased by a factor $\bar{\gamma} < \gamma$ at each step, as long as the error decreases. When the error increases, β_{k+1} is reduced by a factor $\eta > \gamma$ and the upper bound $\bar{\beta}$ is set to the previous value of β that allowed decrease, i.e., β_{k-1} (line 21). We explain this strategy for updating the β_k again in §7.3 .

Remark 9. *We tried to mimic the choice of the β_k's from convex optimization [89] but it did in general perform worse than the simple choice presented here.*

7.3 Heuristic Extrapolation with Restarts (HER)

In this section, we generalize the acceleration method introduced in §7.2 to BCD on solving NTF problems. The content in this section is highly similar to that presented in §7.2 , but expressed in the language of tensors, and the algorithm presented here is in fact slightly different from the one introduced in §7.2 .

As reviewed in §6.3.2 , we have seen that APG and iBPG accelerate block proximal gradient methods by using extrapolation points in the projected gradient step to update each factor matrix. In another line of works, AO (Algorithm 6) was accelerated by using extrapolation between each block update (rather than inside the block update as in APG and iBPG); in other words, each factor matrix is updated by the extrapolation between previous updated factors. In the literature of tensor decomposition, the second type of extrapolation has been used to accelerate alternating least squares algorithms for solving CPD. Those works will be reviewed in §7.3.2 . In the following, we introduce HER - a novel extrapolation scheme that can be categorized into the class of accelerated AO algorithms using extrapolation between block update.

Informal description of the algorithm To achieve this empirical speedup in convergence speed, an extrapolation scheme "à la Nesterov" is used every time a block has been optimized, before switching to another block. The proposed Heuristic Extrapolation with Restarts (HER) algorithm consists of the following steps:

1. Initialize $\mathbf{A}^{(i)} = [\mathbf{a}_1^{(i)}, \ldots, \mathbf{a}_r^{(i)}]$ and pairing (auxiliary) variables $\hat{\mathbf{A}}^{(i)}$ for $i \in [N]$.
2. Loop over the blocks $\mathbf{A}^{(i)}$ ($i \in [N]$):
    a) Update $\mathbf{A}^{(i)}$ by minimizing the NNLS subproblem (6.11), where the other blocks are fixed and take the value of the pairing variables $\hat{\mathbf{A}}^{(j)}$ ($j \neq i$). For example, one can take a

gradient step (see §6.3.1 for more sophisticated strategies). Keep the previous value of $\mathbf{A}^{(i)}$ in memory as $\mathbf{A}^{(i)}_{old}$.

b) Update the pairing variable using extrapolation. For example, for the scheme with $hp = 3$ in §7.2 ,

$$\hat{\mathbf{A}}^{(i)} = \Big[\mathbf{A}^{(i)} + \beta(\mathbf{A}^{(i)} - \mathbf{A}^{(i)}_{old})\Big]_+ .$$

3. If the restart criterion is true (the reconstruction error has increased), reject the extrapolation and reset pairing variables $\hat{\mathbf{A}}^{(i)} = \mathbf{A}^{(i)}$ for $i \in [N]$; otherwise, update $\mathbf{A}^{(i)} = \hat{\mathbf{A}}^{(i)}$ for $i \in [N]$. See §7.3.1 for the details of the restart mechanism.

4. Update the parameter β; see §7.3.1 .

5. If convergence criterion is not met, go back to 2.

This approach has been scarcely studied [5, 15, 86], while extrapolation is a rather well understood method to accelerate both convex and nonconvex single-block descent algorithms. The main novelty of HER is to tackle a nonconvex optimization problem using BCD with extrapolation between the block update, as opposed to inside each block update such as in [76] or after each outer loop as in [86]. This in-between extrapolation comes at almost no additional computational cost.

Remark 10. *We now give some further remarks on a key idea in HER.*

- *As the algorithm illustrates, HER uses the combination of extrapolation with a restart mechanism. When extrapolation is pointing the variable to the right direction, which is where the cost value goes down faster, or where the variable escape the "swamp" (see §6.2.1), we gain acceleration. When the extrapolation is pointing the variable to the wrong direction, which is where the cost value increases, the restart is to safe guard the wrong extrapolation.*

- *The restart criterion has to be computed in all iterations, which is expensive to do so if we use the cost function F as the restart criterion, which basically requires to recompute the function F after the blocks are updated. However, this is only true for other restart mechanisms but not for HER. A key point in HER is that, by design, the restart criterion is cheap to compute as it is partially computed when updating the block variables.*

7.3.1 Details of HER

The pseudo-code of HER is given in Algorithm 8, which is highly similar to Algorithm 7 with $hp = 3$, the only difference it that the last block is also projected. For ease of understanding, Fig.7.5 shows the flow of line 4 to line 8 of Algorithm 8.

In the following, we elaborate on HER.

Update step – line 6 It is clear that Algorithm 8 has the form of an alternating optimization framework in which the key optimization subproblem (7.1) is a NNLS. As reviewed in §6.3.1 , some efficient algorithms for the NNLS subproblem (7.1) include AS, ADMM, Nesterov's accelerated gradient, or A-HALS. The main difference between AO and HER is that HER does not use the latest values of the other blocks $\mathbf{A}^{(j)}$ $(j \neq i)$ but employs the latest values of their extrapolation $\hat{\mathbf{A}}^{(j)}$ $(j \neq i)$. For convenience, we refer to $\Big\{\hat{\mathbf{A}}^{(i)}_k, i \in [N]\Big\}_{k \geq 0}$ as the extrapolation sequence.

Algorithm 8 HER

```
 1: Input: a nonnegative $N$-way tensor
 2: Output: nonnegative factors $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}$.
 3: Initialization: Choose $\beta_0 \in (0,1)$, $\eta \geq \bar{\gamma} \geq \gamma \geq 1$ and 2 sets of initial factor matrices
    $(\mathbf{A}_0^{(1)}, \ldots, \mathbf{A}_0^{(N)})$ and $(\hat{\mathbf{A}}_0^{(1)}, \ldots, \hat{\mathbf{A}}_0^{(N)})$. Set $\bar{\beta}_0 = 1$ and $k = 1$.
 4: for $k = 1, \ldots$ until some criteria is satisfied do
 5:     for $i = 1, \ldots, N$ do
 6:         Update step Let $\mathbf{A}_k^{(i)}$ be an exact/inexact solution of
                $\min_{\mathbf{A}^{(i)} \geq 0} F\left(\hat{\mathbf{A}}_k^{(1)}, \ldots, \hat{\mathbf{A}}_k^{(i-1)}, \mathbf{A}^{(i)}, \hat{\mathbf{A}}_{k-1}^{(i+1)}, \ldots, \hat{\mathbf{A}}_{k-1}^{(N)}\right).$    (7.1)
 7:         Extrapolation step
                $\hat{\mathbf{A}}_k^{(i)} = \left[\mathbf{A}_k^{(i)} + \beta_{k-1}(\mathbf{A}_k^{(i)} - \mathbf{A}_{k-1}^{(i)})\right]_+.$    (7.2)
 8:     end for
 9:     Compute $\hat{F}_k := F\left(\hat{\mathbf{A}}_k^{(1)}, \hat{\mathbf{A}}_k^{(2)}, \ldots, \hat{\mathbf{A}}_k^{(N-1)}, \mathbf{A}_k^{(N)}\right)$.
10:     if $\hat{F}_k > \hat{F}_{k-1}$ then
11:         Set $\hat{\mathbf{A}}_k^{(i)} = \mathbf{A}_k^{(i)}$, $\forall i \in [N]$        % abandon the sequence $\hat{\mathbf{A}}_k^{(i)}$
12:         Set $\bar{\beta}_k = \beta_{k-1}$,   $\beta_k = \beta_{k-1}/\eta$.        % Update $\bar{\beta}$, decrease $\beta$
13:     else
14:         Set $\mathbf{A}_k^{(i)} = \hat{\mathbf{A}}_k^{(i)}$, $\forall i \in [N]$.        % keep the sequence $\hat{\mathbf{A}}_k^{(i)}$
15:         Set $\bar{\beta}_k = \min\{1, \bar{\beta}_{k-1}\bar{\gamma}\}$,   $\beta_k = \min\{\bar{\beta}_{k-1}, \beta_{k-1}\gamma\}$.        % Increase $\bar{\beta}$ and $\beta$
16:     end if
17: end for
```

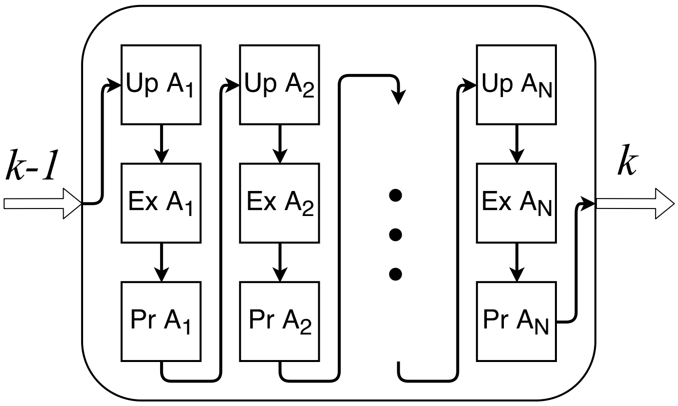


Fig. 7.5. The flow of line 4 to line 8 of Algorithm 8. In the figure, k is the iteration number, “Up" refers to “update”, “Ex" refers to “extrapolate” and “Pr” refers to “project”.

Extrapolation step – line 7 After the update of $\mathbf{A}_k^{(i)}$, the same block of the extrapolation sequence $\hat{\mathbf{A}}_k^{(i)}$ is updated by extrapolating $\mathbf{A}_k^{(i)}$ along the direction $\mathbf{A}_k^{(i)} - \mathbf{A}_{k-1}^{(i)}$, see the extrapolation step (7.2). Note that $\hat{\mathbf{A}}_k^{(i)}$ produced by the extrapolation step (7.2) is always feasible. This extrapolation step corresponds to the case $hp = 3$ in Algorithm 7. It is possible to remove the projection in the extrapolation step (7.2) (i.e. which corresponds to $hp = 2$), but we do not focus on this direction for NTF[3]. On feasibility, $\mathbf{A}_k^{(i)}$ produced by line 6 of Algorithm 8 is always feasible regardless of the feasibility of $\hat{\mathbf{A}}_k^{(i)}$.

The restart mechanism – lines 9-16 After the update-extrapolate process on all the blocks, a restart procedure is carried out to decide whether or not we replace $\mathbf{A}^{(i)}$ $(i \in [N])$ with the extrapolation sequence. Line 14 of Algorithm 8 has the same spirit with the updates (7.6b) and (7.7) where the factor matrices are updated by *the extrapolation between block update.*

It may raise a question why $F(\mathbf{A}_{k-1}^{(1)}, \dots, \mathbf{A}_{k-1}^{(N)})$ does not appear in the restarting condition – line 10. The answer is due to the practicality of the algorithm. As stated in [5], using F as the restart criterion is computationally much more expensive than using the approximate $\hat{F}$. When computing $\hat{F}$, no explicit computation is required; instead, one may reuse already computed components from the updates of $\mathbf{A}^{(N)}$ and $\hat{\mathbf{A}}^{(N)}$. This creates an important reduction of computational complexity. e.g., consider an order-N NTF problem with factor matrices $\{\mathbf{A}^{(i)}\}_{i=1,2,\dots,N}$ with size $I_1 \times r$, $I_2 \times r$, $\dots$ up to $I_N \times r$. Reusing already computed components (such as gradient) in the update of the last block ($\mathbf{A}^{(N)}$, $\hat{\mathbf{A}}^{(N)}$), we can compute $\hat{F}(\hat{\mathbf{A}}^1, \dots, \hat{\mathbf{A}}^{N-1}, A^N)$ under $I_N r^{N-1}$ flops. However, if we compute $F(\mathbf{A}^1, \dots, \mathbf{A}^N)$, it takes $\prod_{i=1}^N I_i$ flops. If $r^N \ll \prod_i I_i$, then such reduction in complexity from $\prod_{i=1}^N I_i$ to $I_N r^{N-1}$ is significant even when N is low. Furthermore, we can always rotate the tensor such that I_N is the mode with the smallest size among all the modes. In fact, computing the cost function naively can be as costly as one block update, and thus using $\hat{F}$ instead of F as the restart criterion is important, since restart using F requires computing the cost function at each iteration, while restart using $\hat{F}$ is much cheaper.

Moreover, note that if the iterates sequence is converging, then the extrapolated sequence also converges to the same limit point. Therefore, since F is a continuous map, if convergence of the iterates is observed then the surrogate cost $\hat{F}$ will asymptotically converge to the same final value as F. Although we did not characterize how fast this convergence happens, this justifies to use $\hat{F}$ as a surrogate at least near a stationary point.

The extrapolation parameters in lines 9-16 The extrapolation weight β_k is computed within the restart mechanism of lines 9-16 of Algorithm 8 , and it is updated using four parameters; see Table 7.1 . In the initialization stage, we set $\bar{\beta}_0 = 1$, pick $\beta_0 \in]0, 1[$, and select η, γ and $\bar{\gamma}$ such that $1 < \bar{\gamma} \leq \gamma \leq \eta$. The parameter $\bar{\beta}$ is called the upper bound parameter for β. This parameter is used to limit the growth of β; see below for more details. The parameter γ is called the (multiplicative) growth rate of β: when the error decreases, β is updated with $\gamma\beta$. Similarly $\bar{\gamma}$ is the (multiplicative) growth rate of $\bar{\beta}$. Finally, η is called the decay rate of β. This value is used to update β with β/η when the error increases. The parameters $\gamma, \bar{\gamma}, \eta$ are fixed constants, while β and $\bar{\beta}$ are updated depending on the restart condition.

The update of β HER updates β_k as

$$\beta_{k+1} \quad = \quad \begin{cases} \beta_k/\eta & \text{if } \hat{F}_{k+1} > \hat{F}_k \\ \min\{\gamma\beta_k, \bar{\beta}_k\} & \text{if } \hat{F}_{k+1} \leq \hat{F}_k \end{cases}, \tag{7.3}$$

which is explained as follows:

[3] In fact, when using the same algorithm for solving CPD, without the nonnegativity constraint, we remove the projection as it is not required.x

Table 7.1: Parameters in the HER scheme

Symbol	Name	Setting	Range	Requires tuning?
β_k	Extrapolation weight	update as (7.3)	$[0, 1]$	Yes for β_0
γ	Growth rate of β	constant	$[\bar{\gamma}, \eta]$	Yes
η	Decay rate of β	constant	$[\gamma, \infty[$	Yes
$\bar{\gamma}$	Growth rate of $\bar{\beta}$	constant	$[1, \gamma]$	Yes
$\bar{\beta}_k$	Upper bound for β	update as (7.4)	$[\beta_k, 1]$	No, $\bar{\beta}_0 = 1$

- If restart occurs, i.e., $\hat{F}_{k+1} > \hat{F}_k$, we assume it is caused by an over-sized β_k (recall that, for $\beta_k = 0$, decrease is guaranteed by the update in line 6) and we shrink the value of β for the next iteration using the decay parameter η as in the update (7.3).
- Otherwise, $\hat{F}_{k+1} \le \hat{F}_k$, we assume β_k can safely be increased. We grow β for the next iteration as $\gamma\beta$. To prevent β to grow indefinitely, we use an upper bound $\bar{\beta}$ as in the update (7.3).

The update of $\bar{\beta}$ HER updates $\bar{\beta}_k$ as follows

$$\bar{\beta}_{k+1} \quad = \quad \begin{cases} \beta_k & \text{if } \hat{F}_{k+1} > \hat{F}_k \\ \min\{\bar{\gamma}\bar{\beta}_k, 1\} & \text{if } \hat{F}_{k+1} \le \hat{F}_k \end{cases}, \tag{7.4}$$

which is explained as follows:

- If there is no restart, i.e., $\hat{F}_{k+1} \le \hat{F}_k$, then $\bar{\beta}$ is increased if $\bar{\beta}$ is smaller than 1.
- Otherwise $\hat{F}_{k+1} > \hat{F}_k$ and $\bar{\beta}_{k+1}$ is set to β_k to prevent β_{k+1} growing larger than β_k too fast in the future. In fact, β_k indicates a too large value for β since the error has increased.

Remark 11. *Let us make a few remarks.*

- *The relationships between the parameters in HER are:*

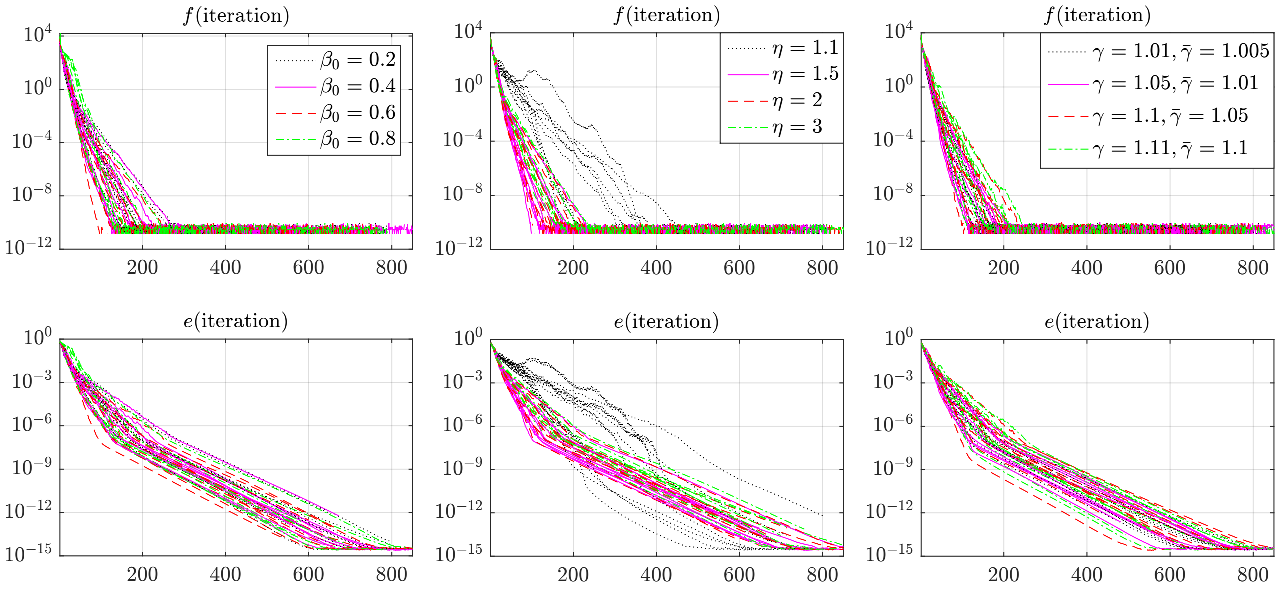
$$0 < \beta_k \le \bar{\beta}_k \le 1 < \bar{\gamma} \le \gamma \le \eta < +\infty. \tag{7.5}$$

  By construction, $\beta_k \le \bar{\beta}_k \le 1$, while $\bar{\gamma} \le \gamma$ ensures that $\bar{\beta}$ increases slower than β, while $\gamma \le \eta$ ensures that β is decreased faster.

- *We observed that HER is more effective if the NNLS subproblems* (7.1) *are solved with relatively high precision. Empirically Fig. 7.17 suggested to use HER with repeated projected gradient steps rather than just a single step. The suffix 50 after the algorithms' name in Fig. 7.17 means that we run 50 iterations for the algorithms to solve the NNLS subproblem* (7.1)*.*

- *A drawback of the HER approach is the parameter tuning. There are 4 parameters to tune: $\beta_0, \gamma, \bar{\gamma}, \eta$. However HER is not too sensitive for reasonable values of the parameters; see Fig. 7.6 for an illustration. So, normally no parameter tuning is needed, even in difficult cases when data are ill-conditioned or rank is high; namely we will use $\beta_0 = 0.5$, $\gamma = 1.05$, $\bar{\gamma} = 1.01$ and $\eta = 1.5$, these values are used in [3].*

- *In the implementation, we initialize $\hat{\mathbf{A}}_0^{(i)} = \mathbf{A}_0^{(i)}, i = 1, \ldots, N$.*



(a) On fix $\gamma, \bar{\gamma}, \eta$. **(b)** On fix $\gamma, \bar{\gamma}, \beta_0$. **(c)** On fix β_0, η.

Fig. 7.6. Comparison of HER with different parameters on the same NTF problems: a rank-10 factorization on noiseless tensors generated by random with size 50 × 50 × 50. For each set of parameters, the decomposition is repeated 10 times over 10 different data tensors and initializations; the top plots representing f display the error of the approximation, and the bottom plots representing e display the distance to the ground truth factors; see §7.5 for details. The default set of parameters are $[\beta_0 = 0.5, \gamma = 1.05, \bar{\gamma} = 1.01, \eta = 1.5]$. The results here show that HER is not sensitive to its parameters as all the curves are not deviating away from each other, except for the case $\eta = 1.1$, suggesting that η should not be too small. Figure copied from [3].

7.3.2 Related works to HER

Here we present two extrapolation schemes similar in spirit with HER, namely Bro, GR and LS.

Extrapolated AO algorithms with Harshman-Bro's sequence Extrapolated AO algorithms can be traced back to the seminal work of Harshman in the 70s [53]. Extrapolation was then seen as a way to speed up the convergence of *Alternating Least Squares* (ALS). The proposed heuristic by Harshman was later revisited and optimized by Bro [15] with convincing empirical speed-ups for computing CPD (i.e., without the nonnegativity constraint). The scheme of Bro is the following heuristic:

$$\begin{aligned} \text{Update:} \quad & \mathbf{A}^{(i)}_{k+\frac{1}{2}} \text{ by solving NNLS subproblem (6.11)}, && (7.6a)\\ \text{Extrapolate:} \quad & \mathbf{A}^{(i)}_{k+1} = \mathbf{A}^{(i)}_{k+\frac{1}{2}} + \left(k^{\frac{1}{h(k)}} - 1\right)\left(\mathbf{A}^{(i)}_{k+\frac{1}{2}} - \mathbf{A}^{(i)}_{k}\right), && (7.6b) \end{aligned}$$

where k is the current iteration index and $h(k)$ is a recursive function so that $h(k+1) = h(k)$ if the error has not increased for more than 4 iterations, $h(k+1) = 1 + h(k)$ otherwise, and $h(1) = 3$. Note that, before moving to the extrapolation step (7.6b), the update step (7.6a) is carried out across all block i. Moreover, no extrapolation is performed in the first few (4 in this paper) iterations because of stability issues.

There is however no particular modification of the Bro's scheme for the nonnegative CPD (i.e. NTF) case. Furthermore, empirically, Bro's accelerated BCD diverges when factorization rank is high. In [3], Bro-AHALS, Bro-ADMM and Bro-Nesterov – the 3 versions of Bro's accelerated methods were implemented, in which we respectively use the same strategy using AHALS (see §6.3.1), ADMM and Nesterov's accelerated gradient method for solving the NNLS subproblem (6.11) inexactly.

Extrapolated AO algorithms with gradient ratio and line search Recently, [86] have considered two heuristic approaches similar to the approach of Bro [15]. The two heuristic approaches follow a two-step framework. In the first step, an update on a variable $\mathbf{x}_k$ is performed, where $\mathbf{x}_k$ is obtained by stacking all the block $\mathbf{A}_k^{(i)}$ into one single vector, and hence it is a "all-at-once" approach. In the second step, the extrapolation coefficient ω_k is computed in two different ways (see below). After ω_k is computed, extrapolation is performed on $\mathbf{x}_k$ as $\mathbf{x}_{k+1} = \mathbf{x}_k + \omega_k(\mathbf{x}_k - \mathbf{x}_{k-1})$. There is no auxiliary sequence in the two approaches.

To compute ω_k, the first approaches, referred to as Gradient Ratio (GR), uses $\omega_k = \frac{\nabla_{\mathbf{x}} F_k}{\nabla_{\mathbf{x}} F_{k-1}}$, where $\nabla_{\mathbf{x}} F_k$ is the gradient of F with respect to $\mathbf{x}$ at iteration k. The second approach, namely Line Search (LS), computes ω_k by minimizing $F(\mathbf{x}_k + \omega(\mathbf{x}_k - \mathbf{x}_{k-1}))$ with with respect to ω.

In the experiments conducted in [3], these three approaches (Bro, GR and LS) were compared to HER. However, following modifications were made so that the Bro, GR and LS have the same algorithmic structure as HER. First, the update of $\mathbf{x}_k$ is performed block wise, that is, one $\mathbf{A}^{(i)}$ at a time. Next, a main modification on the extrapolation step in GR and LS was made as follows. Note that the expression $\mathbf{x}_{k+1} = \mathbf{x}_k + \omega_k(\mathbf{x}_k - \mathbf{x}_{k-1})$ in the original GR and LS means all the block variables are extrapolated with the same "global" extrapolation coefficient. i.e., the extrapolation coefficients for all block $\mathbf{A}^{(i)}$ are the same. Here, the global extrapolation coefficient is "split"into block-specific extrapolation coefficient. i.e., in GR, the extrapolation is performed for all i as

$$\mathbf{A}_{k+1}^{(i)} \quad = \quad \mathbf{A}_{k+\frac{1}{2}}^{(i)} + \frac{\|\nabla_{\mathbf{A}^{(i)}} F_k\|}{\|\nabla_{\mathbf{A}^{(i)}} F_{k-1}\|}(\mathbf{A}_{k+\frac{1}{2}}^{(i)} - \mathbf{A}_k^{(i)}), \tag{7.7}$$

where $\nabla_{\mathbf{A}^{(i)}} F_k$ is the gradient of F w.r.t. block $\mathbf{A}^{(i)}$ at iteration k, and $\mathbf{A}_{k+\frac{1}{2}}^{(i)}$ is the block $\mathbf{A}^{(i)}$ at iteration k just after the update. That is, we extrapolate the block right after the update, as in Bro's approach and in HER. Moreover, the extrapolation step (7.7) uses the ratio between the norm of the gradient of the current block $\mathbf{A}$ and the norm of the gradient of the same block in the last iteration.

The same thing is done on splitting the global extrapolation coefficient into block-specific extrapolation coefficient in LS. i.e., the $\left(k^{\frac{1}{h(k)}} - 1\right)$ in the extrapolation step (7.6b) is replaced by the extrapolation weight parameter ω_k, which is computed by solving a minimization subproblem. Consider the update of the ith block at iteration k, then

$$\omega_k \quad = \quad \underset{\omega}{\operatorname{argmin}}\ F\Big(\mathbf{A}_k^{(1)}, \dots, \mathbf{A}_k^{(i-1)}, \mathbf{A}_k^{(i)} + \omega(\mathbf{A}_k^{(i)} - \mathbf{A}_{k-1}^{(i)}), \mathbf{A}_{k-1}^{(i+1)}, \dots, \mathbf{A}_{k-1}^{(N)}\Big). \tag{7.8}$$

By expanding F in terms of ω, the function F in the problem (7.8) can be expressed as a 2nd-order polynomial in ω, and hence a closed-form solution for ω exists: by forming matrix $\mathbf{B}$ in the form of (6.9) as in (6.5),

$$\begin{aligned}
\omega_k \quad &= \quad \underset{\omega}{\operatorname{argmin}}\ \left\|\mathbf{T}_{[i]} - \Big(\mathbf{A}_k^{(i)} + \omega(\mathbf{A}_k^{(i)} - \mathbf{A}_{k-1}^{(i)})\Big)\mathbf{B}^{(i)\top}\right\|_F^2 \\
&= \quad \underset{\omega}{\operatorname{argmin}}\ \Big\| \underbrace{\mathbf{T}_{[i]} - \mathbf{A}_k^{(i)}\mathbf{B}^{(i)\top}}_{\hat{\mathbf{T}}_{[i]}} + \omega\, \underbrace{(\mathbf{A}_k^{(i)} - \mathbf{A}_{k-1}^{(i)})\mathbf{B}^{(i)\top}}_{\Delta} \Big\|_F^2 \\
&= \quad \underset{\omega}{\operatorname{argmin}}\ \underbrace{\|\Delta\|_F^2}_{a}\omega^2 + \underbrace{2\langle \hat{\mathbf{T}}_{[i]}, \Delta\rangle}_{b}\,\omega + \underbrace{\left\|\hat{\mathbf{T}}_{[i]}\right\|_F^2}_{c} \\
&= \quad \underset{\omega}{\operatorname{argmin}}\ a\omega^2 + b\omega + c.
\end{aligned}$$

Then, we solve for the real roots of this quadratic equation, put the roots into $F(\omega)$ and pick the one that gives the smallest value, we obtain ω_k.

Remark 12. *There are a few remarks on GR and LS.*

- *As stated, the implementations of GR and LS in this paper are different from the original one proposed in [86] where GR and LS use a vectorized format. They solve ω_k in LS approximately using cubic line search in the Poblano toolbox. Here we perform the extrapolation in matrix format as Bro, and solve the problem* (7.8) *exactly. By splitting of the extrapolation coefficient into block extrapolation coefficients, the original GR and LS are improved as the ω_k in the new GR and LS are more adapted to each block variable. As for Bro's accelerated algorithms, we implement in this paper GR-AHALS, GR-ADMM, GR-Nesterov, LS-AHALS, LS-ADMM and LS-Nesterov where we correspondingly use the same strategy as for AHALS (see §6.3.1), ADMM and Nesterov's accelerated gradient method for solving the NNLS subproblem* (6.11).

- *The per-iteration cost in both GR and LS schemes is larger than that of Bro. Both Bro, GR and LS have restart, but Bro's extrapolation scheme* (7.6b) *is basically a constant manipulation, while GR has multiple matrix-matrix multiplications and LS even has to solve a minimization subproblem. In general, the per-iteration cost of the extrapolation step in GR and LS is about one ALS, while Bro's extrapolation cost is negligible.*

- *GR and LS are designed for CPD (no nonnegativity constraint) but not NTF (with nonnegativity constraint), and similar to Bro's approach [15], there is no modification of the GR and LS scheme for the nonnegative decomposition case. These mean there is no guarantee on feasibility of iterates produced by these methods for NTF.*

- *As originally designed for CPD, in [2] several tests were conducted to compare HER accelerated ALS with Bro's accelerated ALS, GR-accelerated ALS and LS-accelerated ALS on CPD problems without the nonnegativity constraint. Results showed that HER is still capable to provide acceleration, and the convergence is faster than the other methods.*

7.3.3 Extension to CPD

The HER framework can also be used to solve unconstrained CPD problems (i.e., NTF without nonnegativity constraint). To do so, we use Algorithm 8 directly with the following small modifications:

- Line 6: We replace the update (7.1) by Alternating Least Squares (ALS) update, which has a closed-form expression as
$$\mathbf{A}^{(i)} = \mathcal{T}_{[i]}\mathbf{B}^{(i)^\dagger},$$
where $\mathbf{B}^{(i)}$ is defined in definition (6.9).

- Line 7: As CPD has no nonnegativity constraint, we remove the projection in the extrapolation step (7.2).

7.4 Experiments on NMF

In this section we show the efficiency of the extrapolation scheme, i.e., Algorithm 7, to accelerate the NMF algorithms ANLS (solving the NNLS subproblems using Active-set methods) and AHALS (solving the NNLS subproblems using AHALS). Following the original naming used in [5], we name the HER accelerated variant of these two methods by adding a prefix "E-" on the name of the algorithm. i.e., E-ANLS and E-A-HALS. All the figures in this section come from [5].

7.4.1 Datasets and Experimental setup

Datasets Datasets that are widely used in NMF literature are used, see Tables 7.2. The image datasets are dense matrices that represent facial images, and the document datasets are sparse matrices.

Table 7.2: Image datasets [5] and text mining datasets [112]. For image datasets, $r = 40$ in all experiments. For text datasets, $r = 20$ in all experiments, nnz is the number of nonzeros, and sparsity is given in %: $100 * \#\text{zero}/(mn)$).

Image datasets

Name	# pixels	m	n
ORL	112×92	10304	400
Umist	112×92	10304	575
CBCL	19×19	361	2429
Frey	28×20	560	1965

Text mining datasets

Name	m	n	nnz	sparsity
classic	7094	41681	223839	99.92
sports	8580	14870	1091723	99.14
reviews	4069	18483	758635	98.99
hitech	2301	10080	331373	98.57
ohscal	11162	11465	674365	99.47
la1	3204	31472	484024	99.52

We also consider two types of synthetic datasets:

- Low-rank synthetic datasets: we generate each entry of $\mathbf{W}$,$\mathbf{H}$ using the uniform distribution in $[0, 1]$ and compute $\mathbf{X} = \mathbf{WH}$. For each experiment, we generate 10 such matrices and report the average results.
- Full-rank synthetic datasets: we generate each entry of $\mathbf{X}$ uniformly at random in [0,1] so that $\mathbf{X}$ is a full rank matrix.

In both cases, we use $m = n = 200$ and $r = 20$.

Experimental setup In all cases, we report the average error over 10 random initializations, where the entries of the initial matrices $\mathbf{W}, \mathbf{H}$ are chosen uniformly at random in the interval $[0, 1]$. To compare the solutions generated by the different algorithms, we report the relative data fitting error (3.2) to which we subtract the lowest relative error obtained by any algorithm with any initialization (denoted $e_{\min}$). Mathematically, given the data $\mathbf{X}$ and the solution $(\mathbf{W}^{(k)}, \mathbf{H}^{(k)})$ obtained at iteration k, we report

$$E(k) = \frac{||\mathbf{X} - \mathbf{W}^{(k)}\mathbf{H}^{(k)}||_F}{||\mathbf{X}||_F} - e_{\min}. \tag{7.9}$$

For the low-rank synthetic datasets, we use $e_{\min} = 0$.

Using $E(k)$ instead of $||\mathbf{X} - \mathbf{W}^{(k)}\mathbf{H}^{(k)}||_F$ has several advantages: (i) it allows to take meaningfully the average results over several datasets, and (ii) it provides a better visualization both in terms of initial convergence and in terms of the quality of the final solutions computed by the different algorithms. The reason is that $E(k)$ converges to 0 for the algorithm that was able to compute the best solution which allows us to use a logarithmic scale. Also, when comparing NMF algorithms that may converge to different local minima, comparing the values of $E(k)$ especially makes sense when looking at the speed of convergence.

Tuning parameters: preliminary experiments Before we compare the two NMF algorithms (ANLS and AHALS) and their extrapolated variants, we run some preliminary experiments to choose reasonable values for the parameter of Algorithm 7 (hp) and the parameters to update β_k.

As we will see, the extrapolation scheme performs rather differently for ANLS (that computes an optimal solution of the subproblems) and AHALS (that computes an approximate solution using a few steps of coordinate descent). It will also perform rather differently depending on the value of hp, while it will less sensitive to the values of β_0, γ, $\bar{\gamma}$ and η as long as these values are chosen in a reasonable range.

In the next section, we run the different variants with the following parameters: $\beta_0 = 0.25, 0.5, 0.75$, $\eta = 1.5, 2, 3$, $(\gamma, \bar{\gamma}) = (1.01, 1.005), (1.05, 1.01), (1.1, 1.05)$. For each experiment, we will not be able to display the curve for each extrapolated variants (there would be too many, 82 in total: 3^4 and the original algorithm). So, for each value of hp, we only display the variant corresponding to the parameters that obtained the smallest final average error (best) and the largest final average error (worst). This will be interesting to observe the sensitivity of Algorithm 7 to the way β_k is updated.

Extrapolated ANLS (E-ANLS) The top two plots of Fig. 7.7 show the evolution of the average of the relative data fitting error (7.9) for the low-rank and full-rank synthetic datasets.

We observe the following:

- For all the values of the parameters, E-ANLS outperforms ANLS.
- For the low-rank synthetic datasets, E-ANLS with $hp = 1$ and well-chosen parameters for the update of β_k (e.g., $\beta_0 = 0.5, \eta = 1.5, \gamma = 1.1$) performs extremely well and is able to identify solutions with very small relative error ($\approx 10^{-8}$ in average). In fact, the original ANLS algorithm would not be able to compute such solutions even within several thousand iterations.
- For the full-rank synthetic datasets, E-ANLS variants with hp equal to 1, 2 or 3 perform similarly although choosing $hp = 1$ allows a slightly faster initial convergence.
- The best value for γ is either 1.05 or 1.10. The best value for η is either 1.5 or 2 (3 being always the worst). The algorithm is not too sensitive to the initial β as it is quickly modified within the iterations but $\beta_0 = 0.25$ clearly provides the worst performance in most cases.

We now perform the same experiment on image and document datasets except with fewer parameters (we do not use $\gamma = 1.01$, $\eta = 3$, $\beta_0 = 0.25$) in order to reduce the computational load. The bottom two plots of Fig. 7.7 show the evolution of the average of the relative data fitting error (7.9) for the image and document datasets.

We observe the following:

- As for synthetic datasets, E-ANLS outperforms ANLS for all the values of the parameters.
- Since we removed the values of the parameters performing worst, the gap between the best and worst variants of E-ANLS is reduced.

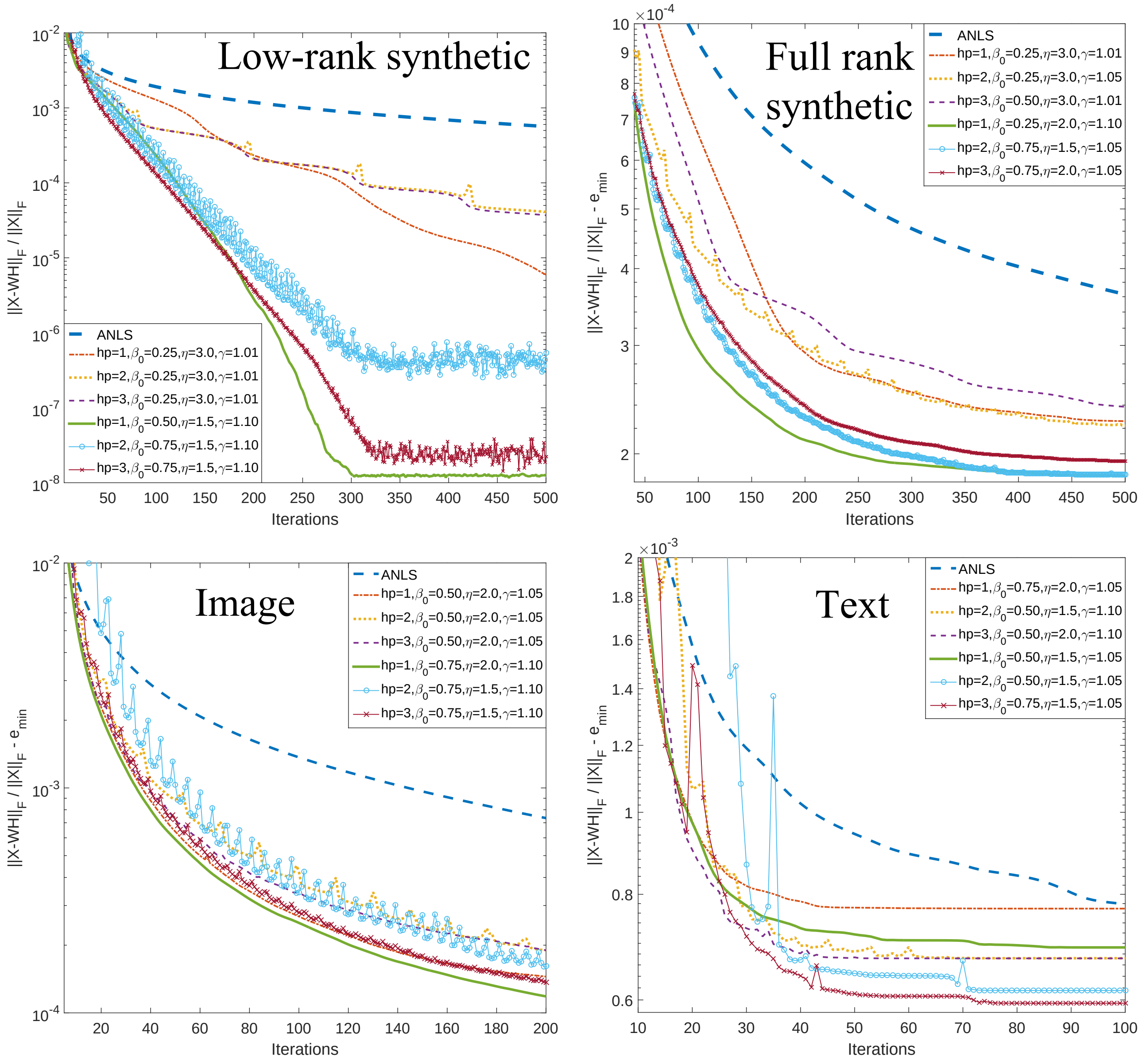


Fig. 7.7. Extrapolation scheme applied to ANLS on synthetic and real datasets. For each value of *hp*, we display the corresponding best and worst performing variant. The curves are the average value of relative data fitting error (7.9) among the different datasets and initializations.

- For the image datasets, the variant with $hp = 1$ performs best although the variants with $hp = 2, 3$ do not perform much worse.
- For the document datasets, the variants with $hp = 2, 3$ perform best (in terms of final error). This can be explained by the fact that NMF problems for sparse matrices are more difficult as there are more local minima with rather different objective function values (see paragraph **Sparse document datasets** for more experiments). So the final error reports the algorithm that found the best solution in most of the 60 cases (6 datasets, 10 initializations per dataset). In terms of speed of convergence, most E-ANLS variants behave similarly (converging within 80 iterations, while ANLS has not converged within 100 iterations).

In the final experiments, we use $\beta_0 = 0.5$, $\eta = 1.5$ and $(\gamma, \bar{\gamma}) = (1.1, 1.05)$ for E-ANLS. We will keep both variants $hp = 1, 3$.

Extrapolated AHALS (E-AHALS) The top two plots of Fig. 7.8 show the evolution of the average of the relative data fitting error (7.9) for the low-rank and full-rank synthetic datasets. For these experiments, we also tested the value $(\gamma, \bar{\gamma}) = (1.005, 1.001)$ (as we will see that smaller value of these parameters perform better).

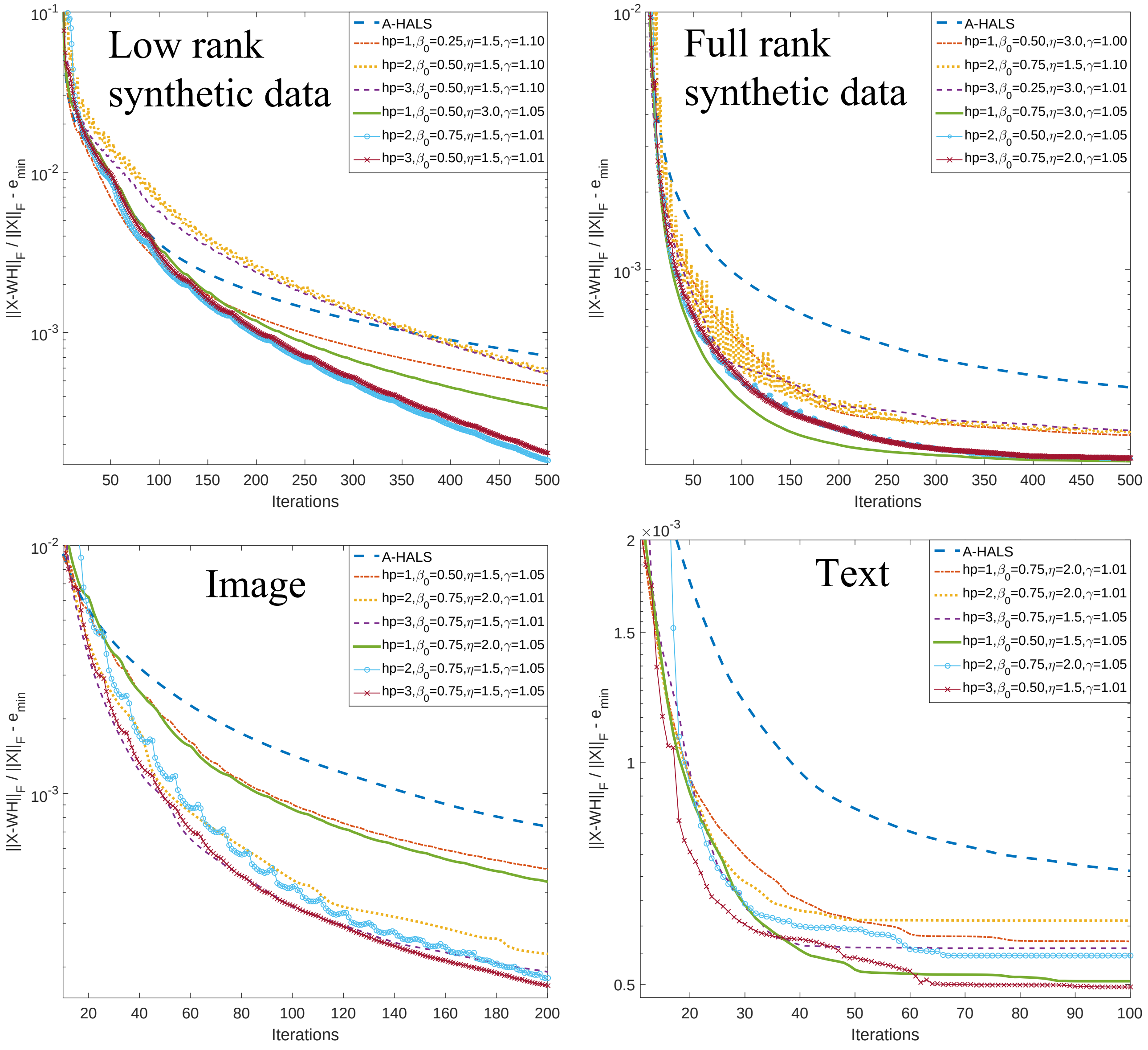


Fig. 7.8. Extrapolation scheme applied to AHALS on synthetic and real datasets. For each value of hp, we display the corresponding best and worst performing variant.

We observe the following:

- For the low-rank synthetic data, with $hp = 2, 3$ and well-chosen parameters for the update of β (e.g., $\beta_0 = 0.50, \eta = 1.5, \gamma = 1.01$), E-AHALS performs much better than AHALS. (Note however that it is not able to find solutions with error as small as E-ANLS within 500 iterations.)
- For the full-rank synthetic data, the variant with $hp = 1$ performs slightly better although the final solutions of the three extrapolated variants have similar error.

- The best value for γ is either 1.01 or 1.05, smaller than for E-ANLS. This can be explained by the fact that AHALS does not solve the NNLS subproblems exactly and the extrapolation should not be as aggressive as for ANLS. As for E-ANLS, E-HALS is not too sensitive to the parameters η and β_0.

We now perform the same experiment on image and document datasets except with fewer parameters (we do not test $\gamma = 1.005, 1.1$, $\eta = 3$, $\beta_0 = 0.25$). The bottom two plots of Fig. 7.8 show the evolution of the relative data fitting error (7.9) for the image and document datasets.

We observe the following:

- For the image datasets, the variant $hp = 1$ performs worse than $hp = 2, 3$ which perform similarly (in terms of speed of convergence).

- For the document datasets, we observe a similar behavior as for ANLS: all extrapolated variants converge much faster than HALS, but converge to different solutions (being in average less than 0.1% away from the lowest relative error).

In the final experiments, we use $\beta_0 = 0.5$, $\eta = 1.5$ and $(\gamma, \bar{\gamma}) = (1.01, 1.005)$ for E-AHALS. We will keep both variants $hp = 1, 3$.

7.4.2 Extensive experiments and comparison of E-ANLS and E-AHALS

We now compare ANLS, AHALS and their extrapolated variants on the same datasets. We also compare these algorithms with the extrapolated alternating projected gradient method for NMF proposed by [109] and referred to as APG-MF.

Synthetic datasets Fig. 7.9 displays the evolution of the average error for the low-rank and full rank synthetic datasets, where the NMF algorithms were run for 20 seconds.

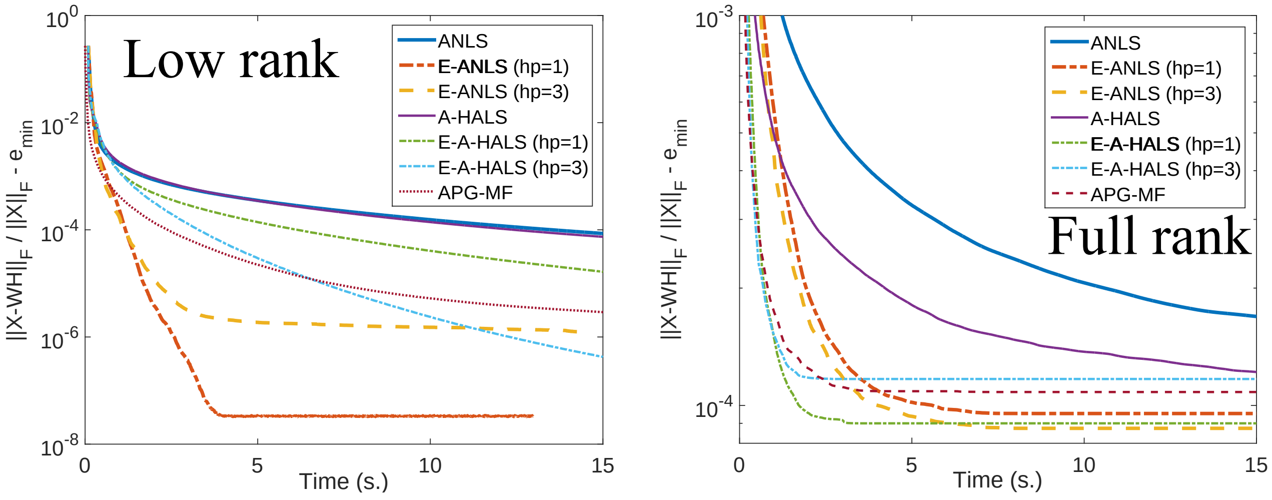


Fig. 7.9. Comparing ANLS, AHALS and their extrapolated variants with APG-MF on synthetic datasets: Average value of the relative data fitting error (7.9).

Tables 7.3 (resp. 7.4) reports the average error, standard deviation and a ranking among the final solutions obtained by the different algorithms for the low-rank (resp. full-rank) synthetic datasets.

Table 7.3: Comparison of the final relative error obtained by the NMF algorithms on the low-rank synthetic datasets: Average error, standard deviation and rankings among the 100 runs (100 datasets). The ith entry of the vector indicates the number of times the algorithm generated the ith best solution. (Observe that all algorithms are able to compute the best solution at least a few times: this happens when they compute an exact solution with $\mathbf{X} = \mathbf{WH}$.)

Algorithm	mean ± std	ranking
ANLS	$5.612\,10^{-5} \pm 7.414\,10^{-5}$	(0, 0, 0, 3, 7, 40, 50)
E-ANLS ($hp = 1$)	$\mathbf{2.618\,10^{-8} \pm 3.657\,10^{-8}}$	(**96**, 4, 0, 0, 0, 0, 0)
E-ANLS ($hp = 3$)	$1.207\,10^{-6} \pm 1.162\,10^{-5}$	(67, 24, 7, 1, 1, 0, 0)
AHALS	$4.547\,10^{-5} \pm 6.299\,10^{-5}$	(1, 0, 0, 4, 10, 41, 44)
E-AHALS ($hp = 1$)	$7.825\,10^{-6} \pm 1.531\,10^{-5}$	(3, 0, 6, 31, 41, 13, 6)
E-AHALS ($hp = 3$)	$1.181\,10^{-7} \pm 3.679\,10^{-7}$	(48, 8, 37, 7, 0, 0, 0)
APG-MF	$2.032\,10^{-6} \pm 5.770\,10^{-6}$	(0, 0, 3, 50, 41, 6, 0)

Table 7.4: Comparison of the final relative error obtained by the NMF algorithms on the full-rank synthetic datasets: Average error, standard deviation and rankings among the 100 runs (10 datasets, 10 initializations each). The ith entry of the vector indicates the number of times the algorithm generated the ith best solution.

Algorithm	mean ± std	ranking
ANLS	$0.423858 \pm 1.183\,10^{-3}$	(4, 9, 9, 12, 22, 21, 23)
E-ANLS ($hp = 1$)	$0.423795 \pm 1.161\,10^{-3}$	(16, 18, 18, 9, 15, 10, 14)
E-ANLS ($hp = 3$)	$\mathbf{0.423787 \pm 1.158\,10^{-3}}$	(**18**, 11, 17, 21, 16, 9, 8)
AHALS	$0.423815 \pm 1.171\,10^{-3}$	(**18**, 12, 11, 18, 13, 13, 15)
E-AHALS ($hp = 1$)	$0.423790 \pm 1.162\,10^{-3}$	(17, 17, 16, 17, 15, 10, 8)
E-AHALS ($hp = 3$)	$0.423817 \pm 1.184\,10^{-3}$	(12, 14, 16, 11, 11, 22, 14)
APG-MF	$0.423808 \pm 1.183\,10^{-3}$	(16, 18, 13, 12, 8, 15, 18)

For low-rank synthetic datasets, these results confirm what we observed previously: E-ANLS ($hp = 1$) is able to significantly accelerate ANLS and obtain solutions with small error extremely fast (in less than 4 seconds). The acceleration of HALS is not as important but it is significant. E-ANLS ($hp = 1$) is able to obtain the best solutions in 96 out of the 100 runs while being always among the two best, while ANLS and AHALS are among the worst ones in most cases. APG-MF never generates the best nor the second best solution.

For full-rank synthetic datasets, we observe that all algorithms obtain a similar final relative error (see Table 7.4), all of them being in average around 0.01% away from the best solution, and there is no clear winner between the extrapolated variants. In fact, there is a priori no reason to believe that an algorithm will converge to a better solution in general since NMF is a difficult nonconvex optimization problem [106]. In terms of speed of convergence, E-AHALS variants converge the fastest (about 3 seconds), followed by APG-MF (about 4 seconds) and the E-ANLS variants (about 8 seconds), while AHALS and ANLS require more than 20 seconds.

Dense image datasets We now run the algorithms for 200 seconds on the 4 image datasets; see Fig. 7.10 which displays the evolution of the average of the relative data fitting error (7.9) for each dataset, and Table 7.2 which compares the final errors obtained by the different algorithms.

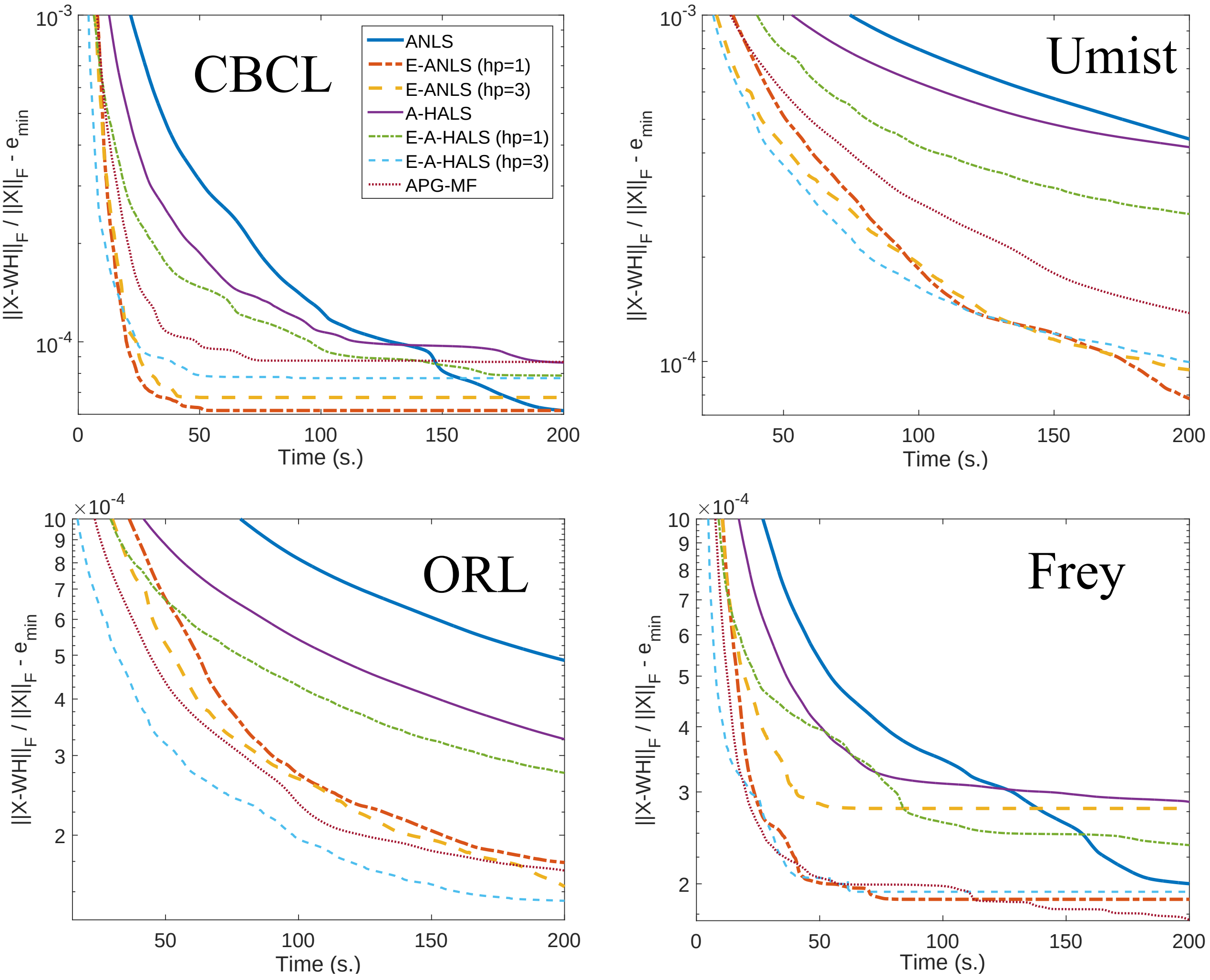


Fig. 7.10. Comparing ANLS, AHALS and their extrapolated variants with APG on image datasets. Average value of the relative data fitting error (7.9) of ANLS, AHALS and their extrapolated variants applied on the 4 image datasets.

We observe the following:

- E-AHALS ($hp = 3$) has the fastest initial convergence speed, followed by E-ANLS variants and APG-MF. As in the preliminary experiments, E-AHALS ($hp = 1$) is able to accelerate AHALS but not as much as E-AHALS ($hp = 3$).
- In terms of final error, there is no clear winner between the extrapolated variants (similarly as for the full-rank synthetic datasets), while ANLS clearly performs the worst.

To conclude, we see that the extrapolation scheme is particularly beneficial to ANLS that is significantly accelerated and even able to outperform E-AHALS in some cases (while AHALS performs in general much better than ANLS, as already pointed out in [46]). Although APG-MF outperforms ANLS (as already observed by [109]) and AHALS, it is in general outperformed by the other extrapolated variants.

Table 7.5: Comparison of the final relative error obtained by the NMF algorithms on the image datasets: Average error, standard deviation and rankings among the 40 runs (4 datasets, 10 initializations each). The ith entry of the vector indicates the number of times the algorithm generated the ith best solution.

Algorithm	mean ± std	ranking
ANLS	$0.110703 \pm 2.964\,10^{-2}$	(3, 3, 3, 5, 5, 8, 13)
E-ANLS ($hp = 1$)	$\mathbf{0.110547 \pm 2.958\,10^{-2}}$	(**9**, 12, 7, 5, 4, 2, 1)
E-ANLS ($hp = 3$)	$0.110570 \pm 2.956\,10^{-2}$	(**9**, 6, 5, 7, 2, 8, 3)
AHALS	$0.110690 \pm 2.956\,10^{-2}$	(1, 4, 4, 2, 3, 13, 13)
E-AHALS ($hp = 1$)	$0.110634 \pm 2.958\,10^{-2}$	(4, 2, 2, 4, 17, 7, 4)
E-AHALS ($hp = 3$)	$0.110552 \pm 2.956\,10^{-2}$	(5, 10, 11, 8, 3, 0, 3)
APG-MF	$0.110559 \pm 2.956\,10^{-2}$	(**9**, 3, 8, 9, 6, 2, 3)

Sparse document datasets We repeat the previous experiments on the 6 document datasets; see Fig. 7.11 which displays the evolution of the relative data fitting error (7.9) for each dataset, and Table 7.6 which compares the final errors obtained by the different algorithms.

Table 7.6: Comparison of the final relative error obtained by the NMF algorithms on the document datasets: Average error, standard deviation and rankings among the 60 runs (6 datasets, 10 initializations each). The ith entry of the vector indicates the number of times the algorithm generated the ith best solution.

Algorithm	mean ± std	ranking
ANLS	$0.850433 \pm 3.186\,10^{-2}$	(5, 3, 12, 6, 9, 11, 14)
E-ANLS ($hp = 1$)	$0.850417 \pm 3.187\,10^{-2}$	(7, 8, 6, 12, 13, 12, 2)
E-ANLS ($hp = 3$)	$0.850324 \pm 3.189\,10^{-2}$	(9, 11, 6, 9, 15, 6, 4)
AHALS	$\mathbf{0.850232 \pm 3.198\,10^{-2}}$	(**15**, 11, 9, 8, 7, 7, 3)
E-AHALS ($hp = 1$)	$0.850287 \pm 3.198\,10^{-2}$	(13, 13, 12, 6, 7, 6, 3)
E-AHALS ($hp = 3$)	$0.850281 \pm 3.204\,10^{-2}$	(**15**, 11, 11, 4, 5, 9, 5)
APG-MF	$0.850471 \pm 3.183\,10^{-2}$	(5, 5, 9, 10, 5, 9, 17)

We observe the following:

- E-AHALS variants have the fastest initial convergence speed converging in average in about 10 seconds, followed by AHALS which sometimes takes much more time to stabilize (e.g., more than 50 seconds for the classic dataset). APG-MF does not converge as fast as E-AHALS variants. E-ANLS variants converge much faster than ANLS but sometimes take more than 30 seconds to stabilize.
- In terms of final error, there is no clear winner although AHALS and E-AHALS ($hp = 3$) gives most of the time the best solution (15 our of 60 cases). APG-MF tends to generate the worst solutions (17 out of the 60 cases) and performs similarly as ANLS in this respect.

For sparse datasets, E-AHALS is the best option for which both variants ($hp = 1, 3$) perform similarly. APG-MF and ANLS and its extrapolated variants are less effective in this case.

Remark 13 (Choice of *hp*)**.** *At this point, we do not have a good theoretical understanding to justify the choice of hp. From the experiments, it is clear that $hp = 2$ should be avoided as it performs in most cases worse than $hp = 1, 3$ while being a more aggressive variant (no projection and extrapolation after the update of both $\mathbf{W}$ and $\mathbf{H}$). Comparing $hp = 1$ and $hp = 3$, there is no clear winner and performance vary from one experiment to another. Understanding this behavior and possibly designing a better strategy (e.g., using a hybridization) is a topic for further research.*

7.4.3 Section summary

The main conclusions are the following:

- In all cases, the extrapolated variants significantly outperform the original algorithms.
- For randomly generated low-rank matrices, E-ANLS, the extrapolated variant of ANLS, allows a significant acceleration, being able to compute solutions with very small relative errors ($\approx 10^{-8}$) in all cases while the other approaches fail to do so.
- For dense datasets, E-ANLS and E-AHALS perform similarly although AHALS performs much better than ANLS. This is interesting: the extrapolated variants allowed ANLS to get back on AHALS.
- For sparse datasets, E-AHALS performs the best and should be preferred to the other variants.
- The extrapolated projected gradient method proposed by [109] and referred to as APG-MF performs well but does not perform as well as the extrapolated variants proposed in this paper.

It would be crucial to understand the extrapolation scheme better from a theoretical point of view. In particular, can we prove convergence to a stationary point as done in [109]? And can we quantify precisely the acceleration like it has been done in the convex case?

7.4.4 Extension to minvol NMF

Finally, here we give an empirical evidence that HER approach also works on minvol NMF problems. Fig.7.12 gives an example on the acceleration made by HER on algorithms solving minvol NMF with $\mathcal{V}_{\text{logdet}}$ on the Samson HSI data. Here we do not tune the HER parameters and use the default set of parameters: $hp = 3, \beta_1 = 0.5$, $\eta = 1.5$, $\gamma = 1.01$ and $\bar{\gamma} = 1.005$. The result shows that HER provides acceleration on the convergence of the sequences $\{\mathbf{W}_k, \mathbf{H}_k\}_{k=1,2,\ldots}$. Similar results can be obtained on other datasets and on other minvol NMF models.

7.5 Experiments on NTF

In this section, we empirically prove the efficacy of HER (Algorithm 8) by extensively test its performance on a rich set of synthetic datasets as well as real datasets. As presented in § 7.3, HER is a scheme to accelerate AO algorithms by using extrapolation between block update; and as such, by using HER, we can derive several different algorithms corresponding to the solver we use for the NNLS subproblem (7.1). We name the algorithms that use AS, ADMM, Nesterov and AHALS for solving NNLS subproblem (7.1) by HER-AS, HER-ADMM, HER-Nesterov and HER-AHALS, respectively. We call HER-AO the set of these algorithms. Table 7.7 lists the algorithms that we

Table 7.7: Algorithms for solving NTF

Algorithms	Reference
HER-AS, HER-ADMM, HER-Nesterov, HER-AHALS	§ 7.3
AO-AS, AO-ADMM, AO-Nesterov, AHALS	§ 6.3.1
GR-ADMM, GR-Nesterov, GR-AHALS	§ 7.3.2
LS-ADMM, LS-Nesterov, LS-AHALS	§ 7.3.2
Bro-ADMM, Bro-Nesterov, Bro-AHALS	§ 7.3.2
APG, iBPG (first-order type methods)	§ 6.3.2

implement and test in our experiments. All the figures in this section come from [3]. In case the legend and the axis label of the figures are too small, we refer the reader to zoom in the figures in the pdf file of this thesis or the pdf file of [3].

7.5.1 Experiment setup

Performance measurement Two important factors in the evaluation of the performance of an algorithm are the data fitting error and the factor fitting error that are computed as follows. We use the value of the objective function

$$f_k := F\left(\mathbf{A}_k^{(1)}, \dots, \mathbf{A}_k^{(N)}\right) \tag{7.10}$$

to represent the fitting error, where F is defined in (6.8). Then, supposing the ground truth factor matrices $\mathbf{A}_{\text{true}}^{(i)}$, $i = 1 \dots, N$ are available, we compute the factor fitting error e_k as

$$e_k := \frac{1}{N}\sum_{i=1}^{N} \frac{\|\text{normalize}(\mathbf{A}_{\text{true}}^{(i)}) - \text{normalize}(\mathbf{A}_k^{(i)})\mathbf{\Pi}\|_F}{\|\text{normalize}(\mathbf{A}_{\text{true}}^{(i)})\|_F}. \tag{7.11}$$

Here normalize$(\cdot)$ is the column-wise normalization step (i.e., the i-th column of normalize$(\mathbf{A})$ is set to $\frac{\mathbf{A}(:,j)}{\|\mathbf{A}(:,j)\|_2}$), and $\mathbf{\Pi}$ is the permutation matrix computed through the Hungarian algorithm. The use of $\mathbf{\Pi}$ is to remove the permutation degree of freedom for matching the columns of $\mathbf{A}^{(i)}$ to the column of $\mathbf{A}_{\text{true}}^{(i)}$, and the use of normalization is to remove the scaling degree of freedom for matching the columns of $\mathbf{A}^{(i)}$ to the column of $\mathbf{A}_{\text{true}}^{(i)}$.

Generate a synthetic data We first generate ground truth factor matrices $\mathbf{A}_{\text{true}}^{(i)} \in \mathbb{R}_+^{I_i \times r}, i = 1, \dots N$ whose entries are sampled from i.i.d. uniform distributions in the interval $[0, 1]$. The tensor $\mathcal{T}^{\text{clean}} \in \mathbb{R}_+^{I_1 \times \dots \times I_N}$ is then constructed from $\mathbf{A}_{\text{true}}^{(i)}$, $i = 1, \dots N$. Finally, we form a synthetic data $\mathcal{T}$ by adding some noise to $\mathcal{T}^{\text{clean}}$, $\mathcal{T} = \max(0, \mathcal{T}^{\text{clean}} + \sigma\mathcal{E})$, where $\sigma \geq 0$ is the noise level, and $\mathcal{E} \in \mathbb{R}^{I_1 \times \dots \times I_N}$ is a tensor whose entries are sampled from a unitary centered normal distribution.

Initialization, number of runs and plots For each run of an algorithm, we use a random initialization, i.e., the initial factor matrices $\mathbf{A}_0^{(i)}$, $i = 1, \dots, N$, are generated by sampling uniform distributions in [0,1]. Note that, testing a specific data tensor, we use the same initialization in one run of all algorithms. We run all the algorithms 20 times with 20 different initializations. We stop one run of an algorithm when the maximum time (which is chosen before running the algorithms) is reached.

In presenting the results, we plot $f - f_{\min}$; and if the ground truth is known, we also plot $e - e_{\min}$. Here $f_{\min}$ and $e_{\min}$ are respectively the minimal value of all the data fitting errors and the factor fitting errors obtained across all algorithms on all runs. In noiseless settings ($\sigma = 0$), exact factorization is possible, so we set $f_{\min} = 0$. To have a better observation of the performance of the algorithms, we plot the curves with respect to both time and iterations [4]. We remark that, "an iteration" for AO algorithms refer to the counter k of the outer loop after all blocks have been updated. Regarding the time evaluation, we record the time stamp for each iteration, and then perform a linear interpolation to synchronize the time curves. Note that such linear interpolation does not reflect 100% truly the real convergence behavior as it is just an linear estimate, but we consider such estimate to be accurate enough.

In our experiment, we emphasize on plotting the median curves of the 20 runs (which are the thick curves in the upcoming figures), because there may be large deviations between different runs.

Solving the NNLS subproblems (6.11) **and** (7.1) When using AS, ADMM, Nesterov's accelerated gradient descent algorithm or AHALS to solve NNLS subproblem (6.11) or (7.1), in our implementation, we terminate the solver when the number of iterations reaches 50 or when $\|\mathbf{A}_s^{(i)} - \mathbf{A}_{s-1}^{(i)}\| \leq 10^{-2}\|\mathbf{A}_0^{(i)} - \mathbf{A}_1^{(i)}\|$, where s is the iteration counter of the solver.

Parameter set up for HER We use the following set of parameters for HER-AO (unless otherwise specified): $\beta_0 = 0.5, \gamma = 1.05, \bar{\gamma} = 1.01, \eta = 1.5$, and extrapolation step (7.2) is used for the extrapolation point.

List of experiments Table 7.8 lists the figures that report our diverse experiments on synthetic data and real data sets. All the experiments have $N = 3$ and the input tensor is dense.

Complete numerical experimental results We refer the interested reader to the appendix of the arXiv paper [3] for all the numerical experiments as reported in Table 7.8. The conclusions remain the same: HER significantly accelerate the convergence of BCD algorithms, while the HER-BCD outperforms both iBPG and APG.

7.5.2 Extensive experiments on synthetic datasets

As listed in Table 7.8, the experiments on synthetic data sets are designed to simulate different kinds of situations that may occur in real applications, which includes: low rank, larger rank, noiseless, noisy, tensor with balanced size (cubic tensor), tensor with unbalanced size (rectangular tensor), and ill-conditioned tensor.

Figure 7.13, 7.14 7.15 and 7.16 strongly affirm that HER-ADMM and HER-AHALS significantly outperform their counterparts AO-ADMM and AHALS in term of both f_k and e_k, where the improvement is often of several orders of magnitude (at least 10^4 in most cases). We observe the same result for HER-AS and HER-Nesterov vs AO-AS and AO-Nesterov.

Compared with APG and iBPG, we observe from Fig. 7.17 that HER-Nesterov outperforms both APG and iBPG in term of f and significantly outperforms them in term of e. From extensive experiments, we observe that HER, the scheme that makes use of the extrapolation between block

[4] We do not report the number of MTTKRP as all the algorithms in the experiments (except for AS) share the same number of MTTKRP (which is N for an tensor with order N), so the performance in terms of number of MTTKRP is contained implicitly in the plot with respect to the iterations.

Table 7.8: List of experiments on NTF.

Fig.	Test description	$[I_1, I_2, I_3, r, \sigma]$
	Synthetic data	
7.13	Cube size, low rank, noiseless	$[50, 50, 50, 10, 0]$
7.13	Unbalanced size, low rank, noiseless	$[150, 10^3, 50, 12, 0]$
7.14	Unbalanced size, larger rank, noiseless	$[150, 10^3, 50, 25, 0]$
7.15	Large cube size, low rank, noisy	$[500, 500, 500, 10, 0.01]$
7.16	Unbalanced size, low rank, noisy, ill-condition	$[150, 10^3, 50, 12, 0.001]$
7.17	HER-AO-gradients vs. APG and iBPG	$[150, 10^3, 50, 10, 0.01]$
7.18	Comparing {HER,Bro,GR,LS}-AHALS	$[50, 50, 50, 10, 0]$ $[150, 10^3, 50, 12, 0.01]$ $[150, 10^3, 50, 25, 0.01]$
	Real data	
7.19	Two HSI images: PaviaU and Indian Pine	$[610, 340, 103, 10]$ $[145, 145, 200, 15]$
7.21	Big data: black-and-white video sequence	$[153, 238, 1.4 \times 10^4, \{10, 20, 30\}]$

update scheme, shows much better performance than APG and iBPG, the accelerated block proximal gradient methods that use Nesterov-type extrapolation inside each block update.

Compared with Bro-AHALS, GR-AHALS and LS-AHALS, Fig. 7.18 shows that our HER-AHALS performs the best in the three experimental settings (only median is plotted here). Note that since the acceleration frameworks Bro, GR and LS are not designed for NTF, it is possible the iterates produced by these frameworks are infeasible. Here we only compare HER-AHALS with Bro- AHALS, GR- AHALS and LS- AHALS; the comparison of these methods where AHALS is replaced with AO-ADMM and AO-ADMM are available in the Appendix of [3], and similar conclusions are drawn, namely that HER outperforms the other accelerations.

7.5.3 Experiments on real datasets: two HSI data

We test the performance of the algorithms on two hyperspectral images (HSI) PaviaU and Indian Pines [5]. They are nonnegative order-3 tensor; PaviaU has size $[610, 340, 103]$ with $r = 10$ and Indian Pines has size $[145, 145, 200]$ with $r = 15$. The r chosen are commonly used in practice.

We perform minimal pre-processing on the raw data: NaN or negative values (if any) are replaced by zero. Hence, it is possible the pre-processed data contains many zeros and being ill-conditioned.

[5] Data available from `http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes`

Figure 7.19 reports the performance of HER-AHALS, HER-ADMM and their counterparts AO-AHALS and AO-ADMM on the two data sets. As there are no ground truth factors, we only show f in the results.

We observe that there are multiple swamps, which are common for real datasets as the data are highly ill-conditioned (the condition numbers of the metricized pre-processed data tensor along all modes are $[593, 642, 1009]$ for Indian Pines and $[944, 462, 8083]$ for PaviaU). Nevertheless, considering the "best case" among the trials, HER-AHALS and HER-ADMM provide solutions with error $10^8 - 10^{10}$ times smaller than the best case of their un-accelerated counterparts. To compare with other algorithms, the readers can view the results in the Appendix of [3]. We observe that iBPG, APG and the AO (AO-AHALS and AO-ADMM) algorithms accelerated by GR, Bro and LS schemes are much slower than our AO (AO-AHALS and AO-ADMM) algorithms accelerated by HER. GR-AO and Bro-AO (for AO being AO-AHALS or AO-ADMM) even sometimes diverge.

7.5.4 Experiments on real datasets: big data, and efficient tensor compression via Tucker format

We test HER-AHALS on the video data of the UCSD Anomaly Dataset [82]. Constructed by combining all the frame images of 70 surveillance video in the dataset, we have a tensor with sizes $153 \times 238 \times 14000$, where the first two modes are the screen resolution and the third mode is the number of frame. The data is shown in Fig.7.20 No pre-processing is performed on the raw data. Data of such size is too big to store in our computer memory, so we perform compression using Tucker decomposition, based on the built-in function from the Tensor toolbox [8]. We compare AHALS and HER-AHALS with $r \in \{10, 20, 30\}$. Results in Fig. 7.21 shows that HER-AHALS performs much better than AHALS.

Tensor compression via Tucker format Now we give the detail on how HER works with Tucker compression. Although there is a long history of using the Tucker model as a compression tool to pre-process big dataset, only recently has been formally discussed that compression does not actually imply transforming the large dataset into a smaller tensor [107]. Given the tensor $\mathcal{T}$, its Tucker format is expressed as:

$$\mathcal{T} = \left(\bigotimes_{p=1}^{N}{}_{a} \mathbf{U}^{(p)} \right) \mathcal{G} \tag{7.12}$$

where $\mathbf{U}^{(p)} \in \mathbb{R}^{n_p \times r_p}$, $\mathcal{G} \in \mathbb{R}^{r_1 \times \dots \times r_N}$ and $\{r_p\}_{p \leq n}$ are inputs integer parameters of the format, sometimes called Tucker ranks [52]. This representation is not unique but still offers a compressed expression of $\mathcal{T}$ thus the name format rather than decomposition.

A typical situation is that of a tensor $\mathcal{T}$ too big to fit in memory, since either too large and dense, or extremely large but sparse. Therefore, a third party may instead provide the data directly in a compact format such as the Tucker format. As Tucker format is in practice an approximation of the real data, the cost function of the NTF problem is modified as follows:

$$F_t(\mathbf{A}^{(1)}, \dots, \mathbf{A}^{(N)}) = \frac{1}{2} \| \left(\bigotimes_{p=1}^{N}{}_{a} \mathbf{U}^{(p)} \right) \mathcal{G} - \left(\bigotimes_{p=1}^{N}{}_{a} \mathbf{A}^{(p)} \right) \mathcal{I}_r \|_F^2. \tag{7.13}$$

On top of the storage gain, there is a huge computational burden ease in using structured representations of the data when computing the MTTKRP. Indeed, the gradient of F_t w.r.t. $\mathbf{A}^{(1)}$ is obtained as follows:

$$\nabla_{\mathbf{A}^{(1)}} F_t = -\mathbf{U}^{(1)}\mathcal{G}_{[1]} \left[\bigodot_{p=2}^{N} (\mathbf{U}^{(p)})^T \mathbf{A}^{(p)} \right] + \mathbf{A}^{(1)} \left[\circledast_{p=2}^{N} (\mathbf{A}^{(p)})^\top \mathbf{A}^{(p)} \right], \tag{7.14}$$

where $\circledast$ is the Hadamard product (elementwise product). Equation (7.14) involves only "cheap" products if Tucker ranks r_p are small compared to the data tensor dimensions n_p. In below, we check that indeed HER-BCD is compatible with accelerating the NCPD using the Tucker format, and this actually opens the door to many problems that could not be tackled with simply HER-BCD, while enhancing at no cost the convergence speed of BCD algorithms for minimizing F_t. This contrasts with usual developments of fast techniques to solve NCPD that typically do not consider other kind of acceleration in conjunction.

As a conclusion for this section, we give some remarks on HER-AO. From our extensive experiments, we observe that HER-AS has inferior performance than others when the data is either big in size, high rank, or ill-conditioned. When the data has small size, all HER-AO algorithms have similar performance, and they all outperform their un-accelerated counterpart algorithms in term of both time and iteration. Among HER-AO algorithms, we highly recommend HER-AHALS for NTF since it shows good performance in all experiments.

7.6 Experiments on unconstrained CPD

In this section we compare HER-ALS to 4 algorithms: the original un-accelerated Alternating Least Squares (ALS), the accelerated ALS using Bro's acceleration (Bro-ALS), the accelerated ALS using line search (LS-ALS), and iBPG. See §7.3.2 for description of Bro's acceleration and line search. All the figures in this section are modified from [2]. In this section, HER-ALS is denoted as herALS in the figures.

We perform tests on 3rd-order tensors using synthetic and real datasets, see Table 7.9. In each experiment, the notation $[I, J, K, r]$ denotes the sizes of the tensor (I, J, K), and the factorization rank r. All experiments are run over 20 random initializations, and we plot the median of the cost value over these 20 trials. Two things to note: all HER-BCD across all experiments use the same set of parameters : $[\beta_0, \gamma, \bar{\gamma}, \eta] = [0.5, 1.05, 1.01, 1.5]$. All the y-axis of the plots is in the form of $F - F_{\min}$, where F is the cost evaluated at all $\mathbf{A}^{(j)}$ and $F_{\min}$ is the minimal cost obtained in the experiment across all initializations and algorithms.

Synthetic datasets Fig. 7.22 shows the result over two experiments on synthetic data. In both balanced and unbalanced cases that were tested, the data tensor is generated as

$$\sum_{q=1}^{r} \mathbf{a}_q^{(1)} \otimes \mathbf{a}_q^{(2)} \otimes \mathbf{a}_q^{(3)} + \mathcal{N},$$

where the ground truth factors $\mathbf{A}^{(j)}$ are sampled from a Gaussian distribution with zero mean and unitary variance. Note that we adjust the condition number of $\mathbf{A}^{(j)}$ to 100 using SVD by replacing the singular values by logarithmic scaled values between 1 and 100. The tensor $\mathcal{N}$ is an additive

Table 7.9: The datasets used for experiments on CPD. The nnz is the number of nonzero after replacing NaN to 0. The sparsity is measured as 100% × #zero/(IJK).

Name	Data	Size $[I, J, K]$	r	nnz	sparsity
Balanced	synthetic	$[50, 50, 50]$	10	-	-
Unbalanced	synthetic	$[150, 1000, 50]$	10	-	-
Wine	real data	$[44, 2700, 200]$	15	2489725	89.52
Indian Pine	real data	$[145, 145, 200]$	16	4205000	0
Blood plasma	real data	$[289, 301, 41]$	3,6,10	2516756	29.43

Gaussian noise with zero mean and variance 0.001. The results show that HER-ALS is the best algorithm among the 5 tested algorithms, and in particular seem to avoid the swamp in which ALS lands in the unbalanced case. LS-ALS, which converges fast in terms of iterations, suffers from higher per-iteration cost.

Real datasets We now show the results on real datasets: Wine[6] (Fig.7.23), HSI data of Indian Pine[7] (Fig.7.24) and Blood plasma[8] (Fig.7.25). Again the curves are the median over 20 initializations. Minimal pre-processing is carried out: NaN value (if any) is replaced by zero, and therefore created some sparsity in the data for the wine data and the blood plasma data. We observe that HER-ALS performs the best, followed by Bro-ALS.

7.7 Chapter summary and perspective

Chapter summary We introduced an acceleration framework, namely HER, for accelerating the algorithms for solving NMF, NTF and CPD. Various numerical experiments demonstrated the effectiveness of HER on accelerating BCD-type algorithms for solving NMF, NTF and CPD problems. We also illustrated that HER can be used to speed up the convergence of minvol NMF algorithm.

Open problem: the theoretical convergence analysis of HER Despite the success of HER in many experiments, the convergence results are still only empirical. The theoretical convergence analysis of HER remains open.

[6] See `http://www.models.life.ku.dk/Wine_GCMS_FTIR` for data description.

[7] `http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes`

[8] See `http://www.models.life.ku.dk/anders-cancer`

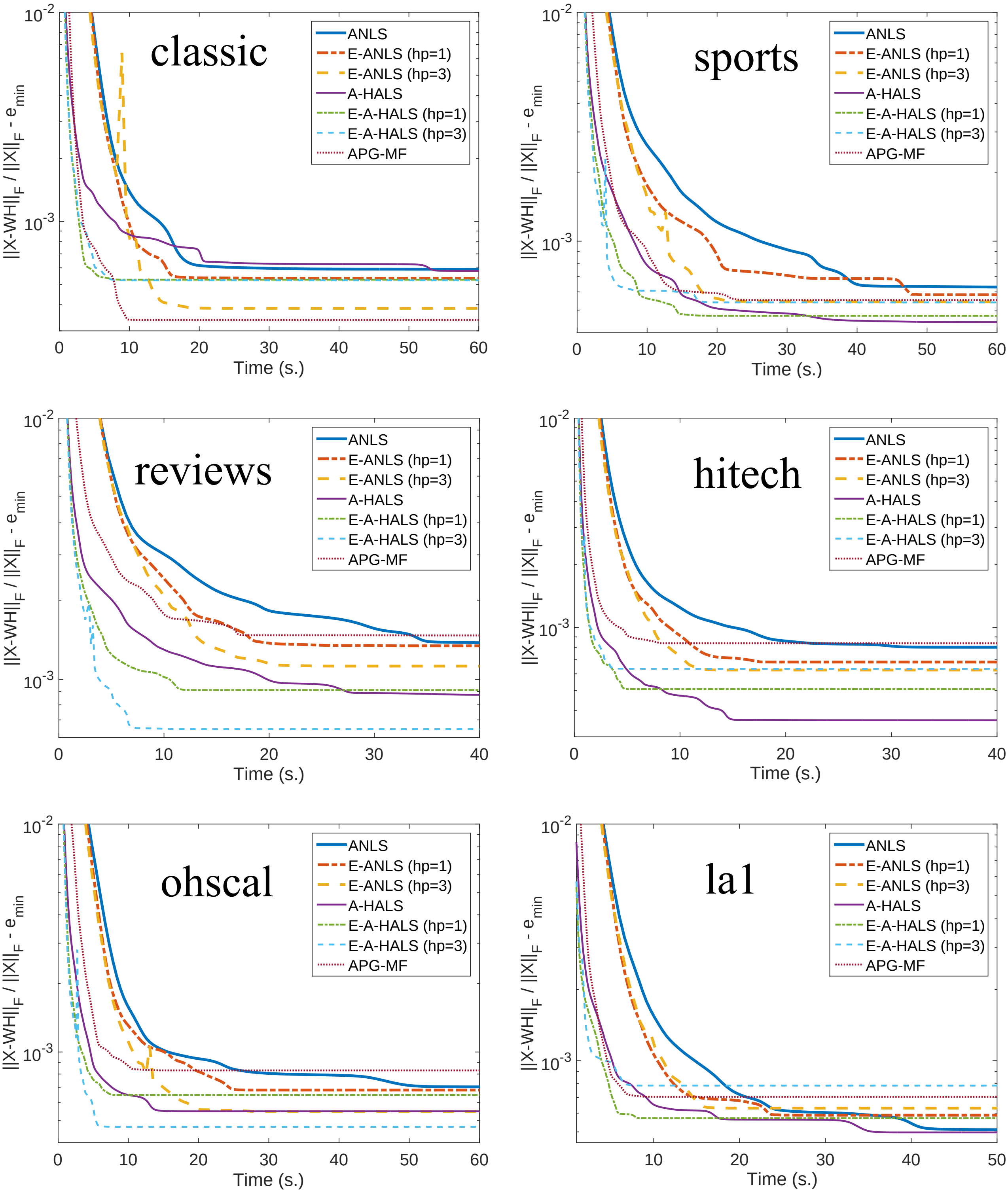


Fig. 7.11. Comparing ANLS, AHALS and their extrapolated variants with APG on text datasets: average value of the relative data fitting error (7.9) of ANLS, AHALS and their extrapolated variants applied on the 6 documents datasets.

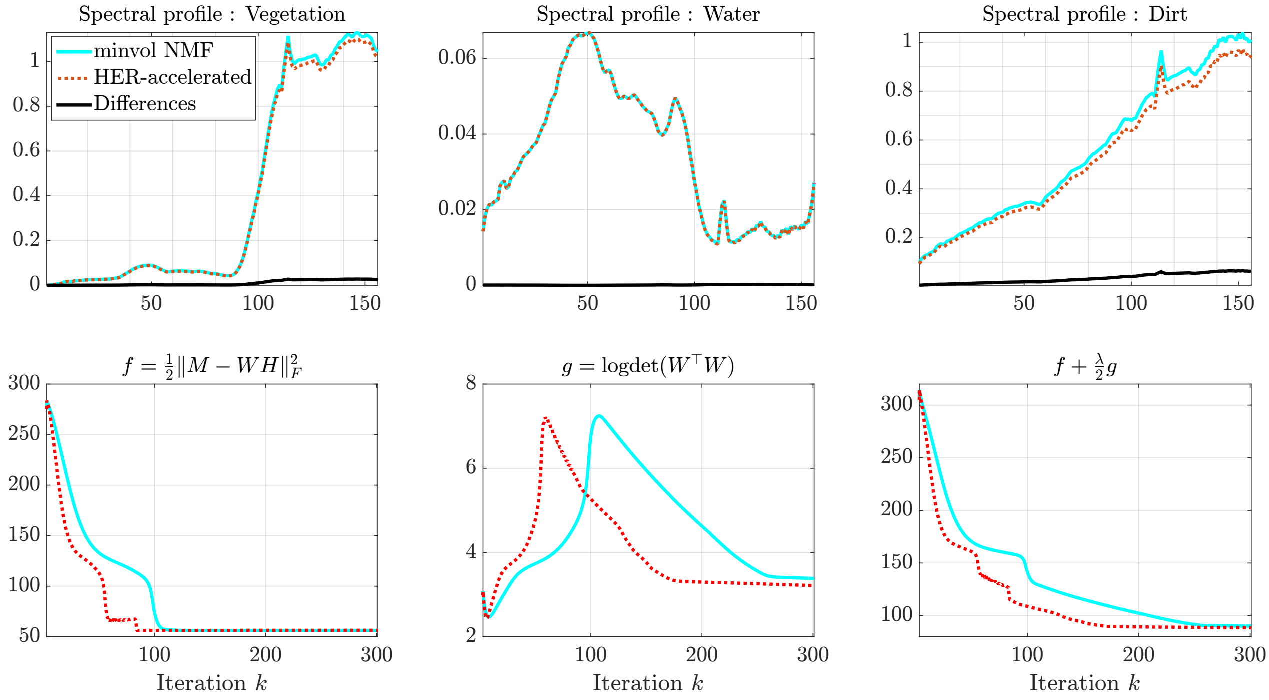


Fig. 7.12. HER on algorithms solving minvol NMF with $\mathcal{V}_{\text{logdet}}$. **Top**: the spectral profile of the endmembers produced by the two algorithms. **Bottom**: the function values, regularizer values versus iteration. Here the dataset Samson (see Fig.3.4 for details) is used. No preprocessing is carried out on the data. In the decompositions, the same rank, same λ and the same initialization is used. For the bottom plots, when plotting the x-axes in computational time, the same results are obtained.

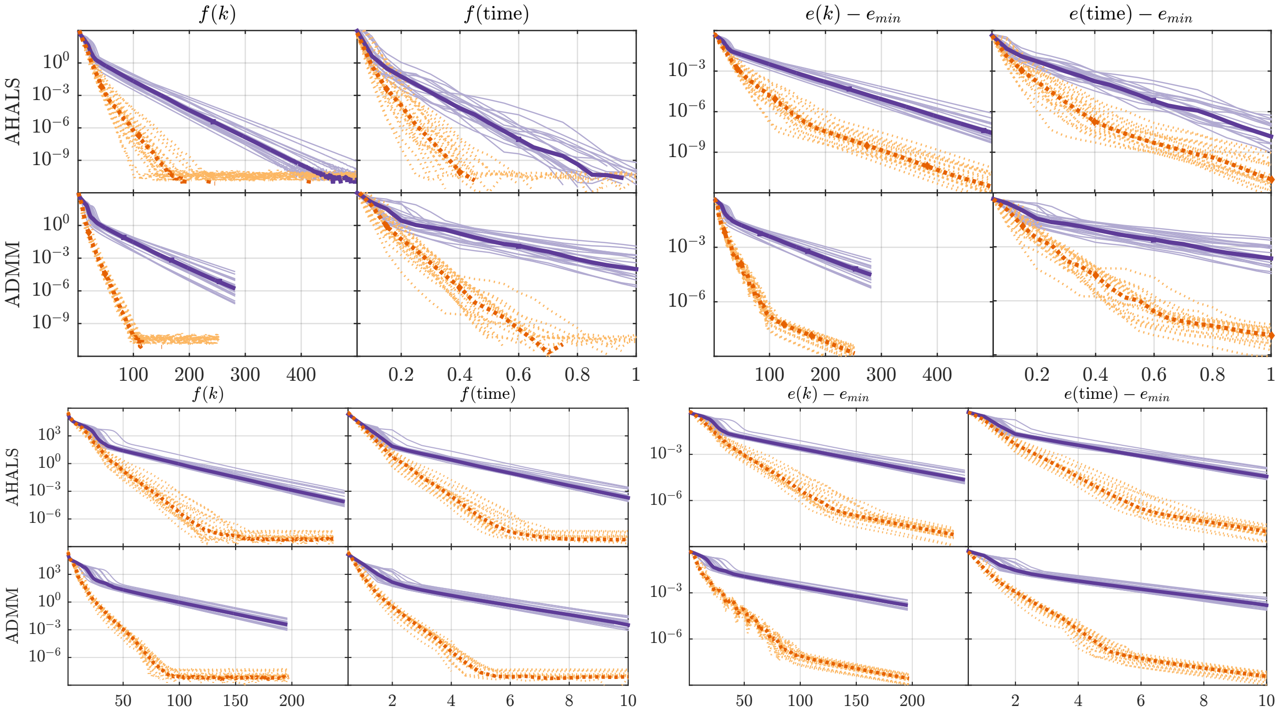


Fig. 7.13. Convergence of algorithms: AHALS and AO-ADMM without HER (solid purple) and with HER (dotted orange), on standard test case (top): $[I_1, I_2, I_3, r] = [50, 50, 50, 10]$ and unbalanced sizes (bottom) $[I_1, I_2, I_3, r] = [150, 10^3, 50, 12]$. The results show that HER improves the convergence significantly, the convergence in both f and e for HER-accelerated methods are already multiple-order of magnitude better than the unaccelerated algorithms. Notice that due to a higher per-outer-iteration cost, ADMM-based algorithms (AO-ADMM and HER-AO-ADMM) run fewer number of outer-iteration than the AHALS-based algorithms. See the Appendix of [3] for the results on other algorithms where we observe a similar behavior.

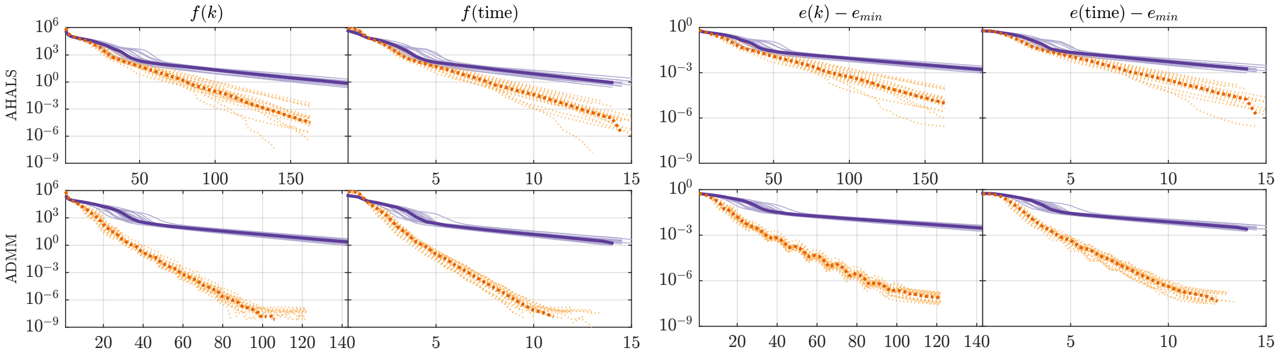


Fig. 7.14. On large rank $[I_1, I_2, I_3, r] = [150, 10^3, 50, 25]$. For the plot set up, see Fig. 7.13. Results show HER improves the convergence speed significantly. See Fig. 7.13 for the plot set up, and the Appendix of [3] for the results on other algorithms.

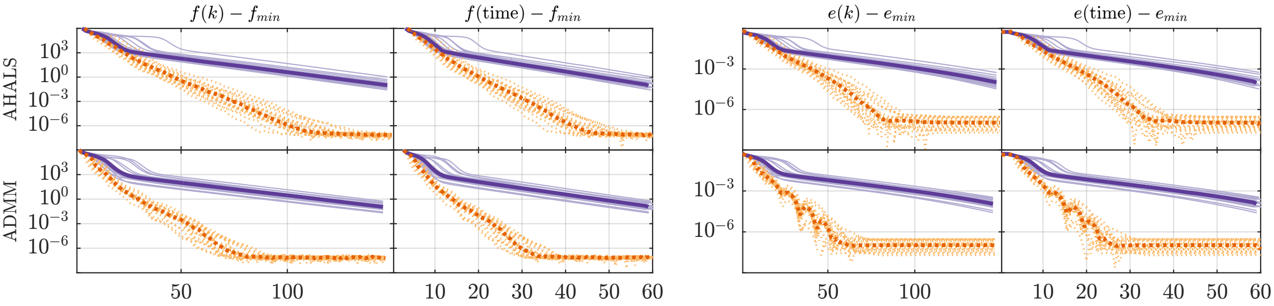


Fig. 7.15. On big and noisy case $I_1 = I_2 = I_3 = 500$, $[r, \sigma] = [10, 0.01]$. Results show HER improves the convergence speed significantly. See Fig. 7.13 for the plot set up, and the Appendix of [3] for the results on other algorithms.

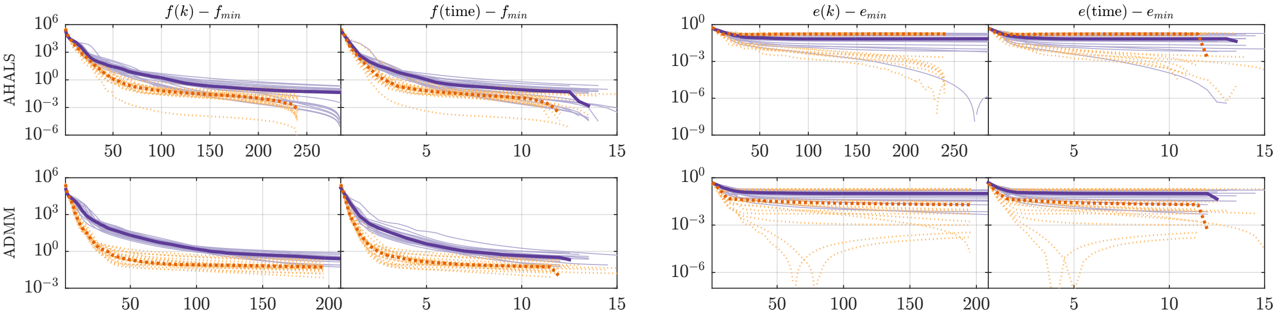


Fig. 7.16. On ill-conditioned case $[I_1, I_2, I_3, r, \sigma] = [150, 10^3, 50, 12, 0.01]$, where $A_i(:,1) = 0.99A_i(:,2) + 0.01A_i(:,1)$ for $i \in \{1, 2, 3\}$. For the plot set up, see Fig. 7.13. Results show HER improves the convergence speed. See Fig. 7.13 for the plot set up, and the Appendix of [3] for the results on other algorithms.

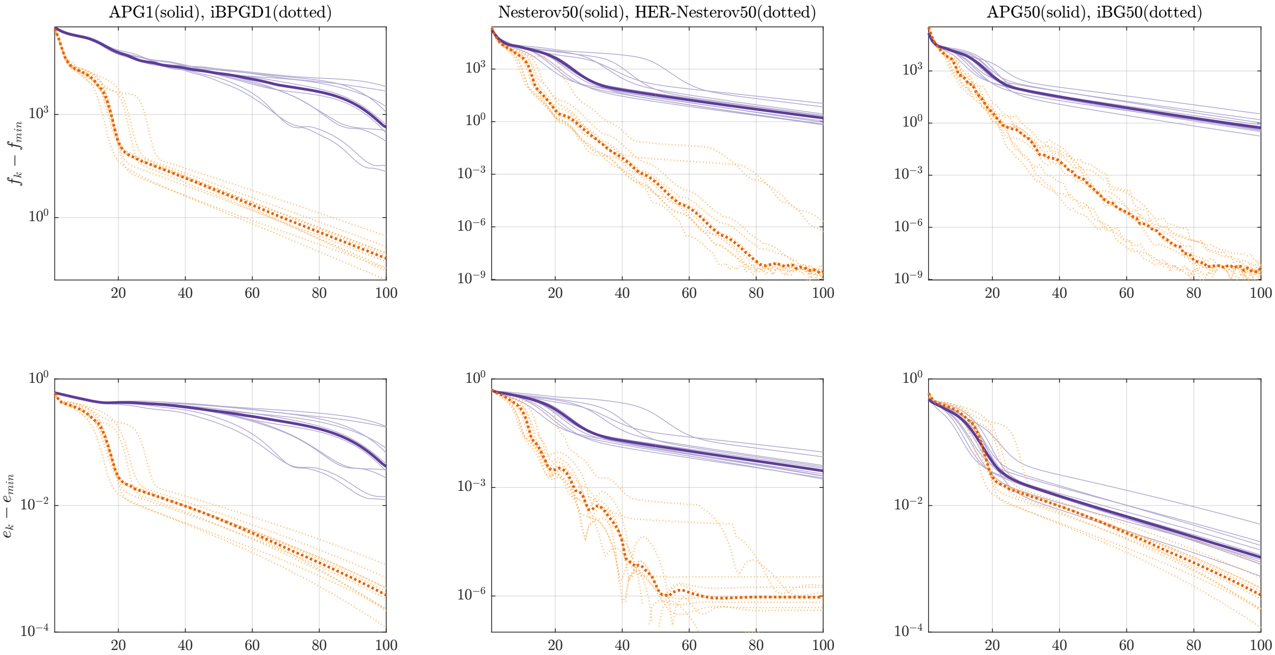
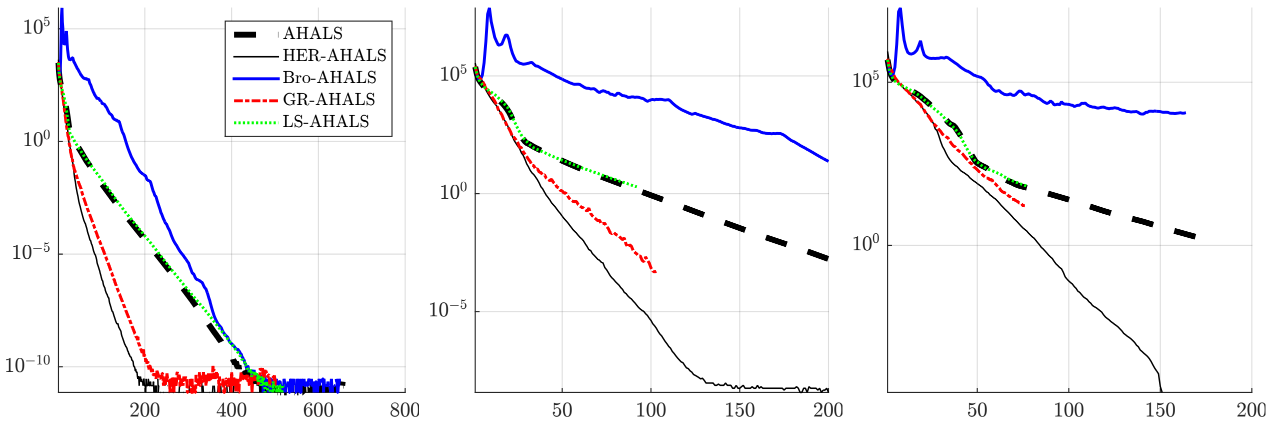
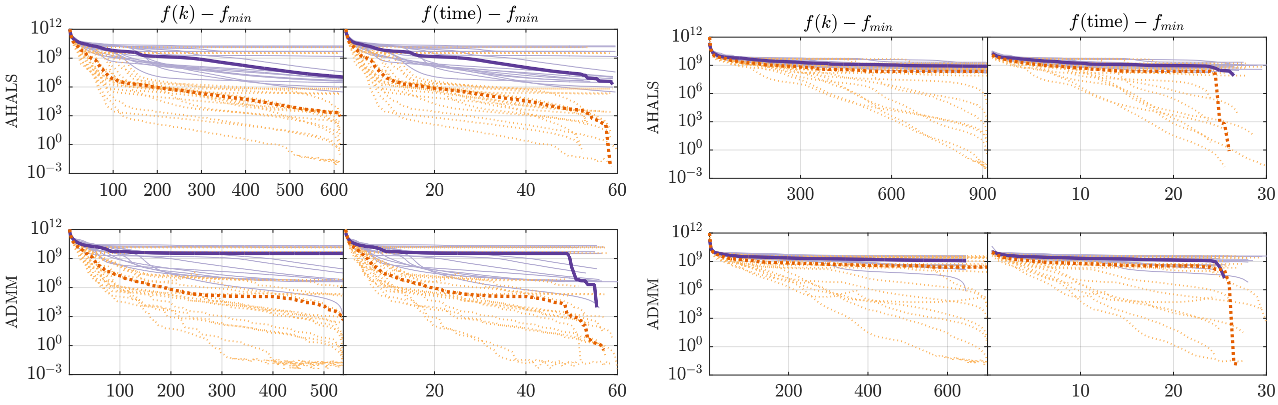


Fig. 7.17. Comparing gradient algorithms on $[I_1, I_2, I_3, r, \sigma] = [150, 10^3, 50, 10, 0.01]$. Suffix number denotes the maximum number of inner iterations. Result shows HER works for both inexact and exact BCD using gradient. Here HER-Nesterov50 and HER-PGD50 are the best algorithms in both f and e. We do not plot the time plot here since they are similar to the iteration plot.



(a) [50, 50, 50, 10, 0] **(b)** $[150, 10^3, 50, 12, 0.01]$ **(c)** $[150, 10^3, 50, 25, 0.01]$

Fig. 7.18. Comparing AHALS with different acceleration frameworks on synthetic datasets on 3 setting of $[I_1, I_2, I_3, r, \sigma]$. The curves are the median in $f(k) - f_{\min}$. The x-axis is the number of iteration, and all algorithm run with same run time limited. Results show HER-AHALS performs better than all other algorithms. LS and GR run less number of iterations due to their larger per-iteration cost. Bro's approach has lower per-iteration cost, but it is even slower than vanilla AHALS. See the Appendix of [3] for more results, and the results of the same experiment with AHALS replaced by AO-ADMM and AO-Nesterov.



(a) On PaviaU dataset. **(b)** On Indian Pines dataset.

Fig. 7.19. The results on HSI data. For the plot set up, see Fig. 7.13. Results show HER improve convergences. See the Appendix of [3] for more results.

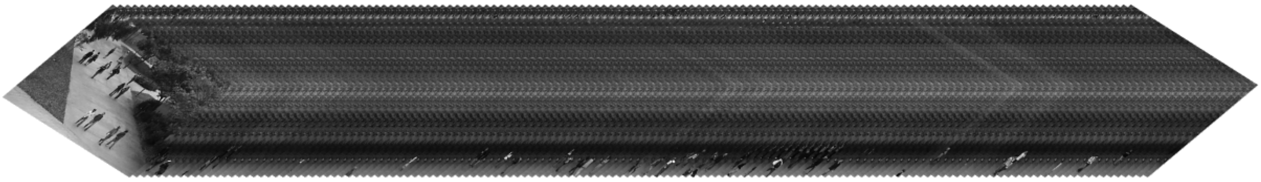
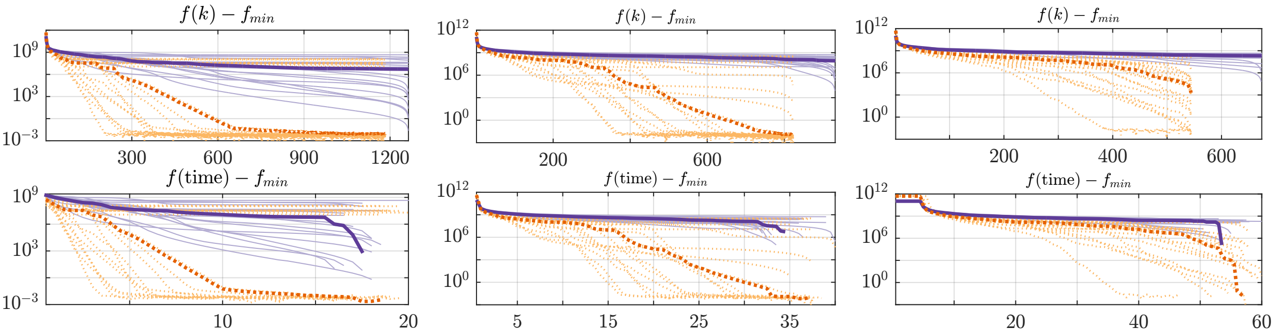


Fig. 7.20. The UCSD Anomaly Dataset.



(a) Case $r = 10$.

(b) Case $r = 20$.

(c) Case $r = 30$.

Fig. 7.21. On video data $[153, 238, 14000]$ for three values of r. Results show HER improve convergence and works well with Tucker-based compression.

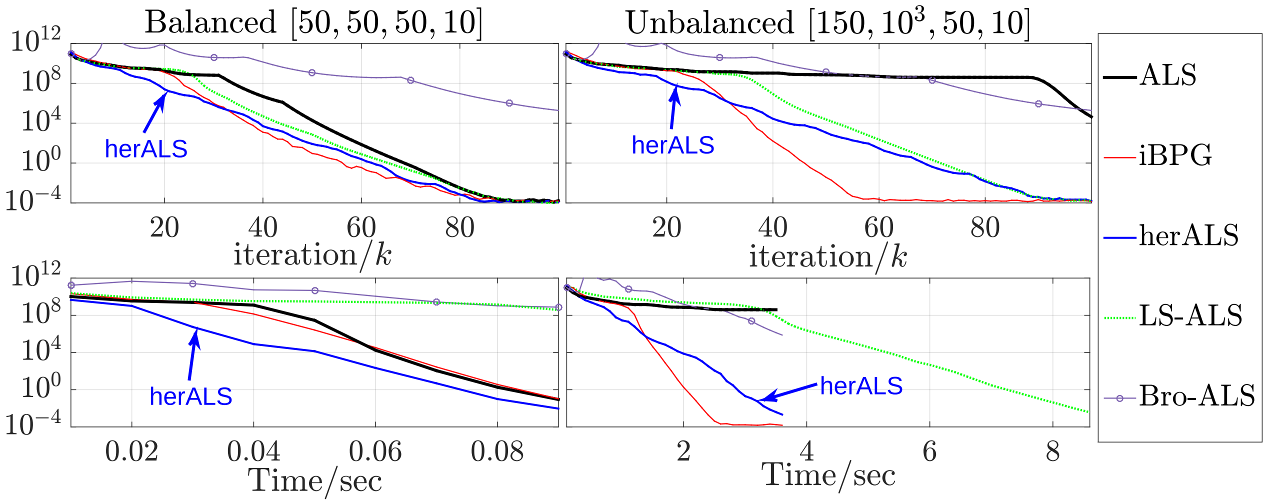


Fig. 7.22. Median error over 20 runs on synthetic datasets plotted against iterations (top) and time (bottom). For the unbalanced case, ALS improves very slowly up to the 90th iteration, illustrated the phenomenon of swamp. HER-ALS do not encounter this issue in this experiment.

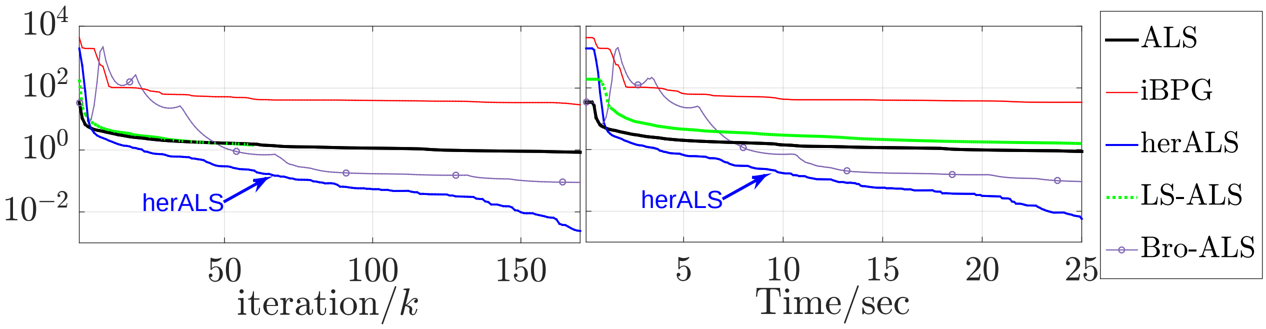


Fig. 7.23. Results on Wine $[44, 2700, 200, 15]$.

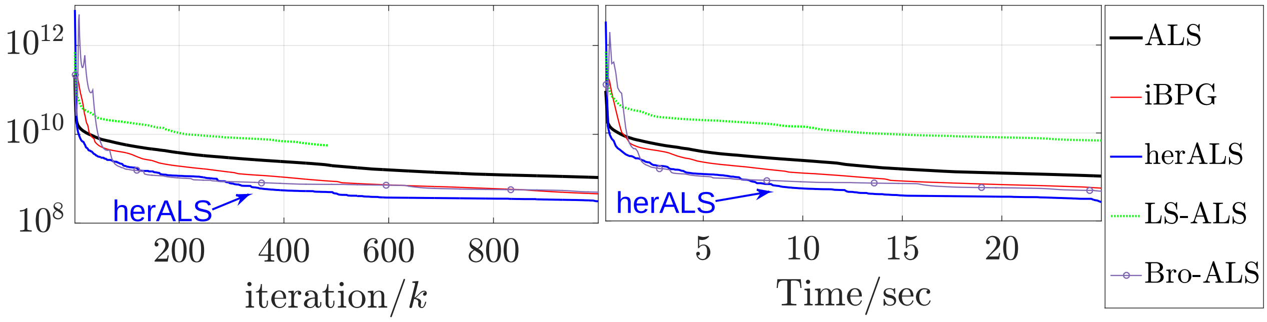


Fig. 7.24. Results on Indian Pines [145, 145, 200, 16].

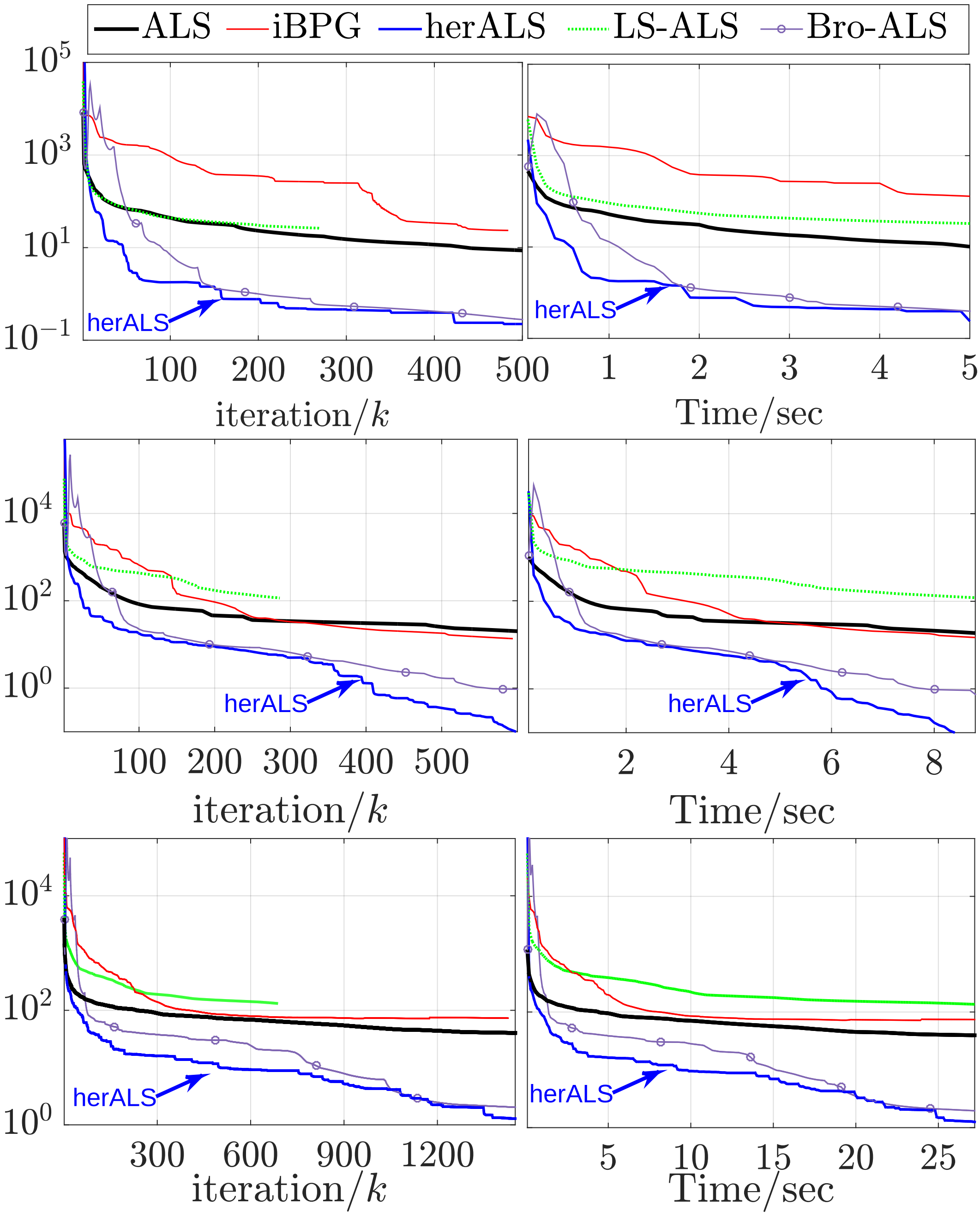


Fig. 7.25. Results on Blood [289, 301, 41] with $r = 3$ (top), $r = 6$ (mid) and $r = 10$ (bottom).

8 Conclusion

Stay hungry. Stay foolish.

Steve Jobs

We conclude the thesis in this chapter. First we summarize the whole thesis.

Chapter 1 We gave a brief introduction of the thesis, and discussed the technical background and the development of NMF for the purpose of this thesis.

Chapter 2 We introduced the minvol NMF models. Minvol NMF with different volume regularizations were investigated, in which we talked about the uses of different volume regularizers, the relationships between these regularizers and some comparisons in the practical setting such as noisy cases and rank deficient cases. We provided an identifiability result on a specific minvol NMF model, namely the noiseless minvol NMF with determinant volume, and showed that the solution of such problem is provably unique under the SSC assumption. Finally, we discussed how to solve the minvol NMF problems. In particular, we solved minvol NMF with $\mathcal{V}_{\text{det}}$ by transforming the objective function into a quadratic form; we solved minvol NMF with $\mathcal{V}_{\text{logdet}}$ with majorization-minimization; and we solved minvol NMF with $\mathcal{V}_{*}$ using a heuristic.

Chapter 3 We introduced Hyperspectral Unmixing (HU) as one of the application where NMF methods shine. We discussed the data format and the performance measurement in HU applications. Then, experimental results showed that min minvol NMF algorithms are superior than others in HU problems on both synthetic and real datasets. In particular, the minvol NMF algorithms are able to outperform both the state-of-the-art separable NMF algorithm `SPA`, and two volume-based methods such as `MVC-NMF` [84] and `RVolMin` [41]. The minvol NMF with determinant volume and log-determinant (logdet) volume are the top two methods among all the tested methods, where the logdet-based one is better than the det-based one.

Chapter 4 We briefly reviewed the audio BSS problem, and discussed how and why NMF can be applied to audio Blind source separation (BSS) problem. By solving the minvol β-divergence NMF model, we demonstrated that it can be used to decompose single channel audio recording of piano music into components correspond to each of the musical notes, showcased the effectiveness of NMF models on audio BSS problems.

Chapter 5 We introduced a new model of NMF, namely Nonnegative Unimodal Matrix Factorization (NuMF). We gave a characterization of the NU set, and proposed a brute-force heuristic to solve NuMF, with acceleration made by a dimension reduction step based on multi-grid method, which is then proved to be able to preserve unimodality. We gave three preliminary results on the identifiability of NuMF under three special cases, and we present numerical experiments on synthetic and real datasets to illustrate the effectiveness of multi-grid and as well as giving empirical evidences to support the findings.

Chapter 6 We briefly discussed the foundation of tensors for the purpose of this thesis, and talked about the Canonical Polyadic Decomposition (CPD) formulation, showed that NMF and NTF problem fall under the class of constrained CPD. Finally we review some algorithms that solve CPD problems in the Block Coordinate Descent (BCD) framework such as the Alternating methods and the Block proximal gradient methods.

Chapter 7 We introduced an acceleration framework, namely Heuristic extrapolation with restarts (HER), for accelerating algorithms for solving NMF, NTF and CPD. We first talked about the idea of extrapolation, then illustrate the original form of HER for accelerating algorithms for solving NMF. Next, we gave a detailed discussion on the HER framework for solving NTF problems. After that, we provide numerical experiments to showcase the effectiveness of HER on accelerating algorithms for solving NMF, NTF and CPD problems, under different experimental situations. Lastly, we illustrated that HER can also be used to speed up the convergence of minvol NMF algorithm.

Summary of contributions In this thesis, we studied models that are theoretical with high practical relevance. As the name suggests, the contributions of this thesis are centered on NMF and NTF over three areas: the modeling aspect, the algorithm aspect and the application aspect, as detailed below.

1. **Modeling aspect: minvol NMF and NuMF** We studied a specific class of NMF problem called *Minimum-Volume NMF* (minvol NMF). This class of NMF problem generalizes another class of NMF called *Separable NMF*. We perform comparison studies on the choice of volume function in minvol NMF. We showed the minvol NMF model with determinant volume is identifiable under the SSC condition, and we argue that such a minvol NMF model is superior than other existing minvol NMF approaches.

   We studied a specific class of NMF problem called *Nonnegative Unimodal Matrix Factorization* (NuMF). We proposed a brute-force heuristic to this problem, with acceleration made by a dimension reduction step based on multi-grid method, which is then proved to be able to preserve unimodality. We also provide three theorems on the identifiability of NuMF in three special cases.

2. **Algorithm aspect: HER acceleration framework and methods for solving minvol NMF and NuMF** As NMF and NTF problems in general are known to be NP-hard, we proposed an efficient but approximate heuristic. We introduced a generic framework, named *Heuristic Extrapolation with Restarts* (HER), on accelerating block coordinate descent type of algorithms in solving NMF and NTF problems. We provided empirical evidence that HER can significantly improve the convergence of various existing BCD algorithms for the problems in various scenarios.

   - The benefits of the HER heuristic are: it can accelerate any BCD algorithm; it extrapolate the variable sequence without increasing the per-iteration computational cost of the algorithm; and the auxiliary extrapolation sequence it produce is always feasible.
   - The main shortcomings of HER are: no theoretical convergence guarantee so far; and HER requires parameter tuning.

   Apart from HER, we provided efficient algorithms for solving minvol NMF and NuMF.

3. **Application aspect: Hyperspectral unmixing, audio source separation and beer CGMS analysis**

   We demonstrated the usefulness of minvol NMF models in two applications: hyperspectral imaging and audio source separation; and we demonstrate the usefulness of NuMF in the application of analyzing the CGMS data of Belgian beers.

Theoretical results We now restate all the new theoretical results developed in this thesis.

- Theorem 2.1.1 **Identifiability of a minvol NMF** Let $\mathbf{M} = \mathbf{WH}$ where $\mathbf{H} \in \mathbb{R}_+^{r\times n}$ satisfies the SSC, $\mathbf{W} \in \mathbb{R}^{m\times r}$ satisfies $\mathbf{W}^\top \mathbf{1}_m = \mathbf{1}_r$ and $\operatorname{rank}(\mathbf{M}) = r$. Then the (exact) solution of the minvol NMF problem (2.8) is essentially unique.

- Theorem 5.2.1 **Restriction operator preserves nonnegative unimodality** Let $\mathbf{x} \in \mathcal{U}_+^m$ and $\mathbf{R} \in \mathbb{R}^{m\times m'}$ with the structure defined in Equation (5.14). Then $\mathbf{y} = \mathbf{Rx} \in \mathcal{U}_+^{m'}$. Furthermore, if $\mathbf{x} \in \mathcal{U}_+^{m,p}$ where p is even, then either $\mathbf{y} \in \mathcal{U}_+^{m,\ p/2-1}$, $\mathbf{y} \in \mathcal{U}_+^{m,\ p/2}$ or $\mathbf{y} \in \mathcal{U}_+^{m,\ p/2+1}$.

- Theorem 5.3.1 **Identifiability of NuMF: strictly disjoint case** Let $\mathbf{M} = \mathbf{WH}$ where: 1. $\mathbf{W}$ is Nu and all the columns have strictly disjoint support, and 2. $\mathbf{H} \in \mathbb{R}_+^{r\times n}$ has $n \geq 1$, $\|\mathbf{h}^i\|_\infty > 0$ for $i \in [r]$. Then the (exact) solution of the NuMF problem (5.3) is essentially unique.

- Theorem 5.3.2 **Identifiability of NuMF: adjacent case** Let $\mathbf{M} = \mathbf{WH}$ where: 1. $\mathbf{W}$ is Nu and adjacent, 2. $\mathbf{H} \in \mathbb{R}_+^{r\times n}$ has $n \geq 1$, $\|\mathbf{h}^i\|_\infty > 0$ for $i \in [r]$, and 3. Independent sensing: for each index pair (j_1, j_2), $j_1, j_2 \in [r]$, $j_1 \neq j_2$ such that the vectors $\mathbf{w}_{j_1}, \mathbf{w}_{j_2}$ satisfy the condition of the vectors $\mathbf{x}, \mathbf{y}$ in Lemma 5.3.2 , the j_1, j_2 rows of $\mathbf{H}$ contains a positive diagonal block $\mathbf{D}$. Then the (exact) solution of the NuMF problem (5.3) is essentially unique.

- Theorem 5.3.3 **On demixing two unequal Nu vectors** Given four non-zero vectors $\mathbf{x}, \mathbf{y}, \mathbf{u}, \mathbf{v}$ in $\mathcal{U}_+^m$. If $\operatorname{supp}(\mathbf{x}) \nsubseteq \operatorname{supp}(\mathbf{y})$, $\operatorname{supp}(\mathbf{x}) \nsupseteq \operatorname{supp}(\mathbf{y})$, $\mathbf{x} = a\mathbf{u} + b\mathbf{v}$ and $\mathbf{y} = c\mathbf{u} + d\mathbf{v}$ with a, b, c, d all nonnegative, then either $\mathbf{u} = \mathbf{x}$, $\mathbf{v} = \mathbf{y}$ or $\mathbf{u} = \mathbf{y}$, $\mathbf{v} = \mathbf{x}$.

Summary of perspectives and future research Now we list the open problems related to the thesis.

- **Robustness of minvol NMF for the $\mathcal{V}_{\mathbf{det}}$ and $\mathcal{V}_{\mathbf{logdet}}$** Unlike SNMF in which there are some robustness results, currently there is no theoretical robustness analysis on minvol NMF model (2.1a). This is a promising but seemingly difficult direction of further research.

- **Identifiability of minvol NMF using $\mathcal{V}_{\mathbf{logdet}}$, $\mathcal{V}_*$ and $\mathcal{V}_{p,\delta}$** The identifiability result developed so far (say, Theorem 2.1.1, and those in [42, 77, 39]) are all based on using the SSC condition (Definition (2.1.3)). The proof technique only works for the det regularizers, but not for the Nuclear norm, Frobenius norm and the smooth Schatten p-norm. How to generalize the identifiability result to minvol models with other regularizers remains open.

- **Theoretical analysis of the automatic model order selection of logdet volume** As stated in §2.1.3 that the logdet regularizer works even in rank deficient case, and it has the ability of automatic model order selection. Furthermore, we see in the experiments in §4.2 that, the minvol KL-NMF automatically set the overestimated components to zero. Currently, the theoretical analysis of such phenomenon remains open. Developing the theoretical understanding of this behavior of the volume regularizer is a interesting research problem.

- **Effective computation for solving minvol NMF with $\mathcal{V}_{\textbf{det}}$ and $\mathcal{V}_*$** When solving the subproblem on $\mathbf{W}$ for minvol NMF with $\mathcal{V}_{\text{det}}$, the solution approach relies on solving the QP subproblem (2.18), in which the coefficients in the QP can be expensive to compute. This makes the iterative algorithm for solving minvol NMF with $\mathcal{V}_{\text{det}}$ expensive. Therefore, it is helpful to develop more efficient algorithms for solving such minvol NMF problem. For the approach to solve minvol NMF with $\mathcal{V}_*$, the current method is just a heuristic with no theoretical convergence guarantee, so an algorithm with theoretical support will be the future work.

- **On handling spectral variability** In §3.3.4, we illustrated the ability of minvol NMF on handling spectral variability, the approach is to use over-estimated r in the factorization, followed by recombination of the components. The result so far is only preliminary, two questions remain open: How to choose an appropriate overestimated rank r? How to recombine the components?

- **Extensions of NMF** We can consider extending the NMF model to handle the violation of the assumptions listed in Table 4.2 . For example, in §4.2 on the experiment on the bell music, we saw that we can relax the assumption that each source (the temporal block) can be well-approximated by a rank-1 component. This leads to the study of Grouped NMF which consider the sub-patterns instead of the rank-1 components to be nonnegative. Other extensions of NMF includes the convolutive NMF, robust NMF, nonlinear NMF and NMF with time-frequency consistency constraint.

- **Automatic music transcription system fully based on NMF technology** As stated in the last section of §4 , it will be interesting to combine minvol KL-NMF with NuMF to build an automatic music transcription system that can capture both the activation and the duration profile of each music note. Such system can be used to convert audio files into sheet music, and it is fully based on NMF technology.

- **Open problems in NuMF** As the result on NuMF are preliminary, many problems remain open.
  - **Restriction operators preserve NU and nonuniform grid** Currently we only show a special class of restriction operator preserves NU. Showing a larger class of restriction operators preserve NU is still open. Furthermore, currently the restriction performs a uniform grid on scaling down the problem. It will be interesting to investigate the use of nonuniform grid which is more adapted to the data structure on searching p.
  - **General identifiability** Identifiability of NuMF in general remains open.

- **Theoretical convergence analysis of HER** Despite the success of HER in many experiments, the convergence results are still only empirical. The theoretical convergence analysis of HER remains open.

Bibliography

[1] Andersen Ang, Jeremy Cohen, and Nicolas Gillis. “Accelerating approximate nonnegative canonical polyadic decomposition using extrapolation”. In: *XXVIIeme Colloque GRETSI.* 2019.

[2] Andersen Man Shun Ang, Jeremy Cohen, Le Thi Khanh Hien, and Nicolas Gillis. “Extrapolated Alternating Algorithms for Approximate Canonical Polyadic Decomposition”. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE. 2020.

[3] Andersen Man Shun Ang, Jeremy E. Cohen, Nicolas Gillis, and Le Thi Khanh Hien. “Accelerating Block Coordinate Descent for Nonnegative Tensor Factorization”. In: *arXiv e-prints* (Jan. 2020).

[4] Andersen Man Shun Ang and Nicolas Gillis. “Volume regularized non-negative matrix factorizations”. In: *2018 9th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS).* IEEE. 2018, pp. 1–5.

[5] Andersen Man Shun Ang and Nicolas Gillis. “Accelerating nonnegative matrix factorization algorithms using extrapolation”. In: *Neural computation* 31.2 (2019), pp. 417–439.

[6] Andersen Man Shun Ang and Nicolas Gillis. “Algorithms and comparisons of nonnegative matrix factorizations with volume regularization for hyperspectral unmixing”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* (2019).

[7] Mário César Ugulino Araújo, Teresa Cristina Bezerra Saldanha, Roberto Kawakami Harrop Galvao, Takashi Yoneyama, Henrique Caldas Chame, and Valeria Visani. “The successive projections algorithm for variable selection in spectroscopic multicomponent analysis”. In: *Chemometrics and Intelligent Laboratory Systems* 57.2 (2001), pp. 65–73.

[8] Brett Bader and Tamara Kolda. “Efficient MATLAB computations with sparse and factored tensors”. In: *SIAM Journal on Scientific Computing* 30.1 (2008), pp. 205–231.

[9] Heinz Bauschke and Patrick Combettes. *Convex analysis and monotone operator theory in Hilbert spaces.* Springer, 2011.

[10] Amir Beck. *First-order methods in optimization.* Vol. 25. SIAM, 2017.

[11] Michael Berry, Murray Browne, Amy Langville, Paul Pauca, and Robert Plemmons. “Algorithms and applications for approximate nonnegative matrix factorization”. In: *Computational statistics & data analysis* 52.1 (2007), pp. 155–173.

[12] José Bioucas-Dias, Antonio Plaza, Gustavo Camps-Valls, Paul Scheunders, Nasser Nasrabadi, and Jocelyn Chanussot. “Hyperspectral remote sensing data analysis and future challenges”. In: *IEEE Geoscience and remote sensing magazine* 1.2 (2013), pp. 6–36.

[13] Dheeraj Bokde, Sheetal Girase, and Debajyoti Mukhopadhyay. “Matrix factorization model in collaborative filtering algorithms: A survey”. In: *Procedia Computer Science* 49 (2015), pp. 136–146.

[14] John Brewer. “Kronecker products and matrix calculus in system theory”. In: *IEEE Transactions on Circuits and Systems* 25.9 (1978), pp. 772–781.

[15] Rasmus Bro. “Multi-way Analysis in the Food Industry: Models, Algorithms, and Applications”. PhD thesis. University of Amsterdam, 1998.

[16] Rasmus Bro and Nicholaos Sidiropoulos. “Least squares algorithms under unimodality and non-negativity constraints”. In: *Journal of Chemometrics: A Journal of the Chemometrics Society* 12.4 (1998), pp. 223–247.

[17] Jian-Feng Cai, Emmanuel Candès, and Zuowei Shen. “A singular value thresholding algorithm for matrix completion”. In: *SIAM Journal on optimization* 20.4 (2010), pp. 1956–1982.

[18] Tsung-Han Chan, Wing-Kin Ma, ArulMurugan Ambikapathi, and Chong-Yung Chi. “A simplex volume maximization framework for hyperspectral endmember extraction”. In: *IEEE Transactions on Geoscience and Remote Sensing* 49.11 (2011), pp. 4177–4193.

[19] Junting Chen and Urbashi Mitra. “Unimodality-constrained matrix factorization for non-parametric source localization”. In: *IEEE Transactions on Signal Processing* 67.9 (2019), pp. 2371–2386.

[20] Andrzej Cichocki, Rafal Zdunek, and Shun-ichi Amari. “Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization”. In: *International Conference on Independent Component Analysis and Signal Separation*. Springer. 2007, pp. 169–176.

[21] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[22] Jeremy Cohen. “About notations in multiway array processing”. In: *arXiv 1511.01306* (2015).

[23] Pierre Comon, Xavier Luciani, and De Almeida Andre. “Tensor decompositions, alternating least squares and other tales”. In: *Journal of Chemometrics* 23.7-8 (2009), pp. 393–405.

[24] Laurent Condat. “Fast projection onto the simplex and the l1 ball”. In: *Mathematical Programming* 158.1-2 (2016), pp. 575–585.

[25] Maurice Craig. “Minimum-volume transforms for remotely sensed data”. In: *IEEE Transactions on Geoscience and Remote Sensing* 32.3 (1994), pp. 542–552.

[26] Margaret Daube-Witherspoon and Gerd Muehllehner. “An iterative image space reconstruction algorthm suitable for volume ECT”. In: *IEEE transactions on medical imaging* 5.2 (1986), pp. 61–66.

[27] Vin De Silva and Lek-Heng Lim. “Tensor rank and the ill-posedness of the best low-rank approximation problem”. In: *SIAM Journal on Matrix Analysis and Applications* 30.3 (2008), pp. 1084–1127.

[28] Anthony Degleris and Nicolas Gillis. “A Provably Correct and Robust Algorithm for Convolutive Nonnegative Matrix Factorization”. In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 2499–2512.

[29] Ralph DeMarr. “Nonnegative matrices with nonnegative inverses”. In: *Proceedings of the American Mathematical Society* 35.1 (1972), pp. 307–308.

[30] Karthik Devarajan. “Nonnegative matrix factorization: an analytical and interpretive tool in computational biology”. In: *PLoS Comput Biol* 4.7 (2008), e1000029.

[31] Julien Dewez and Francois Glineur. “Lower bounds on the nonnegative rank using a nested polytopes formulation”. In: *ESANN 2020, 28th European Symposium on Artificial Neural Networks-Computational Intelligence and Machine Learning.* 2020.

[32] Maryam Fazel. “Matrix rank minimization with applications”. PhD thesis. Stanford University, 2002.

[33] Maryam Fazel, Haitham Hindi, and Stephen Boyd. “Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices”. In: *Proceedings of the 2003 American Control Conference, 2003.* Vol. 3. IEEE. 2003, pp. 2156–2162.

[34] Attila Felinger. *Data analysis and signal processing in chromatography.* Elsevier, 1998.

[35] Olivier Fercoq and Peter Richtárik. “Accelerated, parallel, and proximal coordinate descent”. In: *SIAM Journal on Optimization* 25.4 (2015), pp. 1997–2023.

[36] Cédric Févotte and Jérôme Idier. “Algorithms for nonnegative matrix factorization with the β-divergence”. In: *Neural computation* 23.9 (2011), pp. 2421–2456.

[37] Julien Flamant, Sebastian Miron, and David Brie. “Quaternion Non-Negative Matrix Factorization: Definition, Uniqueness, and Algorithm”. In: *IEEE Transactions on Signal Processing* 68 (2020), pp. 1870–1883.

[38] Gerald B Folland. *Real analysis: modern techniques and their applications.* Vol. 40. John Wiley & Sons, 1999.

[39] Xiao Fu, Kejun Huang, and Nicholas Sidiropoulos. “On identifiability of nonnegative matrix factorization”. In: *IEEE Signal Processing Letters* 25.3 (2018), pp. 328–332.

[40] Xiao Fu, Kejun Huang, Nicholas Sidiropoulos, and Wing-Kin Ma. “Nonnegative Matrix Factorization for Signal and Data Analytics: Identifiability, Algorithms, and Applications.” In: *IEEE Signal Process. Mag.* 36.2 (2019), pp. 59–80.

[41] Xiao Fu, Kejun Huang, Bo Yang, Wing-Kin Ma, and Nicholas Sidiropoulos. “Robust volume minimization-based matrix factorization for remote sensing and document clustering”. In: *IEEE Transactions on Signal Processing* 64.23 (2016), pp. 6254–6268.

[42] Xiao Fu, Wing-Kin Ma, Kejun Huang, and Nicholas Sidiropoulos. “Blind separation of quasi-stationary sources: Exploiting convex geometry in covariance domain”. In: *IEEE Transactions on Signal Processing* 63.9 (2015), pp. 2306–2320.

[43] Nicolas Gillis. “Successive nonnegative projection algorithm for robust nonnegative blind source separation”. In: *SIAM Journal on Imaging Sciences* 7.2 (2014), pp. 1420–1450.

[44] Nicolas Gillis. “The why and how of nonnegative matrix factorization”. In: *Regularization, optimization, kernels, and support vector machines* (2014), pp. 257–291.

[45] Nicolas Gillis. *Nonnegative Matrix Factorization.* In preparation, to be published by SIAM.

[46] Nicolas Gillis and Francois Glineur. “Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization”. In: *Neural computation* 24.4 (2012), pp. 1085–1105.

[47] Nicolas Gillis and Stephen Vavasis. “Fast and robust recursive algorithms for separable nonnegative matrix factorization”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.4 (2013), pp. 698–714.

[48] Xue Gong, Martin J Mohlenkamp, and Todd R Young. “The optimization landscape for fitting a rank-2 tensor with a rank-1 tensor”. In: *SIAM Journal on Applied Dynamical Systems* 17.2 (2018), pp. 1432–1477.

[49] Bryan Grenfell, Ottar Bjornstad, and Barbel Finkenstadt. “Dynamics of measles epidemics: scaling noise, determinism, and predictability with the TSIR model”. In: *Ecological monographs* 72.2 (2002), pp. 185–202.

[50] Bob Grone, Charles Johnson, Marques De Sa, and Henry Wolkowicz. “Improving Hadamard’s inequality”. In: *Linear and Multilinear Algebra* 16.1-4 (1984), pp. 305–322.

[51] Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan. “NeNMF: An optimal gradient method for nonnegative matrix factorization”. In: *IEEE Transactions on Signal Processing* 60.6 (2012), pp. 2882–2898.

[52] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*. Vol. 42. Springer.

[53] Richard Harshman. “Foundations of the PARAFAC procedure: Model and conditions for an “explanatory” multi-mode factor analysis”. In: *UCLA Work. Pap. Phon.* 16 (Nov. 1970), pp. 1–84.

[54] Le Thi Khanh Hien, Nicolas Gillis, and Panagiotis Patrinos. “Inertial block mirror descent method for non-convex non-smooth optimization”. In: *The 37th International Conference on Machine Learning (ICML)*. 2020.

[55] Kejun Huang, Nicholas Sidiropoulos, and Athanasios Liavas. “A flexible and efficient algorithmic framework for constrained matrix and tensor factorization”. In: *IEEE Transactions on Signal Processing* 64.19 (2016), pp. 5052–5065.

[56] Kejun Huang, Nicholas Sidiropoulos, and Ananthram Swami. “Non-negative matrix factorization revisited: Uniqueness and algorithm for symmetric decomposition”. In: *IEEE Transactions on Signal Processing* 62.1 (2013), pp. 211–224.

[57] Zhengyu Huang, Aimin Zhou, and Guixu Zhang. “Non-negative matrix factorization: A short survey on methods and applications”. In: *International Symposium on Intelligence Computation and Applications*. Springer. 2012, pp. 331–340.

[58] Hans-Joachim Hubschmann. *Handbook of GC/MS*. Wiley Online Library.

[59] Tao Jiang and Bala Ravikumar. “Minimal NFA problems are hard”. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 1991, pp. 629–640.

[60] Ramakrishnan Kannan, Mariya Ishteva, and Haesun Park. “Bounded matrix factorization for recommender system”. In: *Knowledge and information systems* 39.3 (2014), pp. 491–511.

[61] Jingu Kim, Yunlong He, and Haesun Park. “Algorithms for nonnegative matrix and tensor factorizations: A unified view based on block coordinate descent framework”. In: *Journal of Global Optimization* 58.2 (2014), pp. 285–319.

[62] Tamara Kolda and Brett Bader. “Tensor decompositions and applications”. In: *SIAM review* 51.3 (2009), pp. 455–500.

[63] Wim P Krijnen, Theo K Dijkstra, and Alwin Stegeman. “On the non-existence of optimal solutions and the occurrence of “degeneracy” in the Candecomp/Parafac model”. In: *Psychometrika* 73.3 (2008), pp. 431–439.

[64] Jonathan Le Roux, Hirokazu Kameoka, Nobutaka Ono, and Shigeki Sagayama. "Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency". In: *International Conference on Digital Audio Effects.* 2010.

[65] Jonathan Le Roux, Nobutaka Ono, and Shigeki Sagayama. "Explicit consistency constraints for STFT spectrograms and their application to phase reconstruction". In: *SAPA@ INTERSPEECH.* 2008, pp. 23–28.

[66] Jonathan Le Roux, Emmanuel Vincent, Yuu Mizuno, Hirokazu Kameoka, Nobutaka Ono, and Shigeki Sagayama. "Consistent Wiener filtering: Generalized time-frequency masking respecting spectrogram consistency". In: *International Conference on Latent Variable Analysis and Signal Separation.* Springer. 2010, pp. 89–96.

[67] Daniel Lee and Sebastian Seung. "Algorithms for non-negative matrix factorization". In: *Advances in neural information processing systems.* 2001, pp. 556–562.

[68] Hyekyoung Lee and Seungjin Choi. "Group nonnegative matrix factorization for EEG classification". In: *Artificial Intelligence and Statistics.* 2009, pp. 320–327.

[69] Hyekyoung Lee, Yong-Deok Kim, Andrzej Cichocki, and Seungjin Choi. "Nonnegative tensor factorization for continuous EEG classification". In: *International journal of neural systems* 17.04 (2007), pp. 305–317.

[70] Augustin Lefevre. "Dictionary learning methods for single-channel source separation". PhD thesis. École normale supérieure Paris-Saclay, 2012.

[71] Valentin Leplat, Andersen Man Shun Ang, and Nicolas Gillis. "Minimum-volume rank-deficient nonnegative matrix factorizations". In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE. 2019, pp. 3402–3406.

[72] Valentin Leplat, Nicolas Gillis, and Man Shun Ang. "Blind Audio Source Separation with Minimum-Volume Beta-Divergence NMF". In: *IEEE Transactions on Signal Processing* (2020).

[73] Valentin Leplat, Nicolas Gillis, Xavier Siebert, and Andersen Man Shun Ang. "Séparation aveugle de sources sonores par factorization en matrices positives avec pénalité sur le volume du dictionnaire". In: *XXVIIeme Colloque GRETSI.* 2019.

[74] Adrian Lewis and Hristo Sendov. "Nonsmooth analysis of singular values. Part I: Theory". In: *Set-Valued Analysis* 13.3 (2005), pp. 213–241.

[75] Tao Li and Chris Ding. *Nonnegative Matrix Factorizations for Clustering: A Survey.* 2013.

[76] Athanasios Liavas, Georgios Kostoulas, Georgios Lourakis, Kejun Huang, and Nicholas Sidiropoulos. "Nesterov-based alternating optimization for nonnegative tensor factorization: Algorithm and parallel implementation". In: *IEEE Transactions on Signal Processing* 66.4 (2017), pp. 944–953.

[77] Chia-Hsiang Lin, Wing-Kin Ma, Wei-Chiang Li, Chong-Yung Chi, and ArulMurugan Ambikapathi. "Identifiability of the simplex volume minimization criterion for blind hyperspectral unmixing: The no-pure-pixel case". In: *IEEE Transactions on Geoscience and Remote Sensing* 53.10 (2015), pp. 5530–5546.

[78] Chih-Jen Lin. "Projected gradient methods for nonnegative matrix factorization". In: *Neural computation* 19.10 (2007), pp. 2756–2779.

[79] Jin-Xing Liu, Dong Wang, Ying-Lian Gao, Chun-Hou Zheng, Yong Xu, and Jiguo Yu. "Regularized non-negative matrix factorization for identifying differentially expressed genes and clustering samples: a survey". In: *IEEE/ACM transactions on computational biology and bioinformatics* 15.3 (2017), pp. 974–987.

[80] David Luenberger and Yinyu Ye. *Linear and nonlinear programming, Fourth edition.* Springer, 2015.

[81] Wing-Kin Ma, José M Bioucas-Dias, Tsung-Han Chan, Nicolas Gillis, Paul Gader, Antonio Plaza, ArulMurugan Ambikapathi, and Chong-Yung Chi. "A signal processing perspective on hyperspectral unmixing: Insights from remote sensing". In: *IEEE Signal Processing Magazine* 31.1 (2014), pp. 67–81.

[82] Vijay Mahadevan, Weixin Li, Viral Bhalodia, and Nuno Vasconcelos. "Anomaly detection in crowded scenes". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* IEEE. 2010, pp. 1975–1981.

[83] Stéphane Mallat. *A wavelet tour of signal processing: The Sparse Way.* Elsevier, 2009.

[84] Lidan Miao and Hairong Qi. "Endmember extraction from highly mixed data using minimum volume constrained nonnegative matrix factorization". In: *IEEE Transactions on Geoscience and Remote Sensing* 45.3 (2007), pp. 765–777.

[85] Ben Mitchell and Donald Burdick. "Slowly converging PARAFAC sequences: swamps and two-factor degeneracies". In: *Journal of Chemometrics* 8.2 (1994), pp. 155–168.

[86] Drew Mitchell, Nan Ye, and Hans De Sterck. "Nesterov acceleration of alternating least squares for canonical tensor decomposition". In: *arXiv 1810.05846* (2018).

[87] Karthik Mohan and Maryam Fazel. "Iterative reweighted algorithms for matrix rank minimization". In: *Journal of Machine Learning Research* 13.Nov (2012), pp. 3441–3473.

[88] Alexandru Németh and Sandor Zoltan Németh. "Order isotonicity of the metric projection onto a closed convex cone". In: *arXiv 1602.04743* (2016).

[89] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course.* Vol. 87. Springer Science & Business Media, 2013.

[90] Ludivine Nus, Sébastian Miron, and David Brie. "An ADMM-based algorithm with minimum dispersion constraint for on-line unmixing of hyperspectral images". preprint. Nov. 2019. URL: `https://hal.archives-ouvertes.fr/hal-02346998`.

[91] Brendan O'donoghue and Emmanuel Candès. "Adaptive restart for accelerated gradient schemes". In: *Foundations of computational mathematics* 15.3 (2015), pp. 715–732.

[92] Myriam Rajih, Pierre Comon, and Richard Harshman. "Enhanced line search: A novel method to accelerate PARAFAC". In: *SIAM journal on matrix analysis and applications* 30.3 (2008), pp. 1128–1147.

[93] Meisam Razaviyayn, Mingyi Hong, and Zhi-Quan Luo. "A unified convergence analysis of block successive minimization methods for nonsmooth optimization". In: *SIAM Journal on Optimization* 23.2 (2013), pp. 1126–1153.

[94] Yaroslav Shitov. "A short proof that NMF is NP-hard". In: *arXiv 1605.04000v1* (2016).

[95] Yaroslav Shitov. “A universality theorem for nonnegative matrix factorizations”. In: *arXiv 1606.09068* (2016).

[96] Yaroslav Shitov. “The nonnegative rank of a matrix: Hard problems, easy solutions”. In: *SIAM Review* 59.4 (2017), pp. 794–800.

[97] Nicholas Sidiropoulos, Lieven De Lathauwer, Xiao Fu, Kejun Huang, Evangelos Papalexakis, and Christos Faloutsos. “Tensor decomposition for signal processing and machine learning”. In: *IEEE Transactions on Signal Processing* 65.13 (2017), pp. 3551–3582.

[98] Paris Smaragdis, Cedric Fevotte, Gautham Mysore, Nasser Mohammadiha, and Matthew Hoffman. “Static and dynamic source separation using nonnegative factorizations: A unified view”. In: *IEEE Signal Processing Magazine* 31.3 (2014), pp. 66–75.

[99] Suvrit Sra and Inderjit Dhillon. *Nonnegative matrix approximation: Algorithms and applications.* Computer Science Department, University of Texas at Austin, 2006.

[100] Richard Stanley. “Log-concave and unimodal sequences in algebra, combinatorics, and geometry”. In: *Annals of the New York Academy of Sciences* 576.1 (1989), pp. 500–535.

[101] Gilbert Strang. “Linear algebra and its applications, Thomson Learning”. In: *Inc., London* (1988).

[102] Joel Tropp. *Literature survey: Nonnegative matrix factorization.* Tech. rep. University of Texas at Asutin, 2003.

[103] Paul Tseng. “Convergence of a block coordinate descent method for nondifferentiable minimization”. In: *Journal of optimization theory and applications* 109.3 (2001), pp. 475–494.

[104] Konstantin Usevich, Valentin Emiya, David Brie, and Caroline Chaux. “Characterization of finite signals with low-rank STFT”. In: *2018 IEEE Statistical Signal Processing Workshop (SSP).* IEEE. 2018, pp. 393–397.

[105] Christophe Vanderaa. “Development of a state-of-the-art pipeline for high throughput analysis of gas chromatography - mass spectrometry data”. Master Thesis. Katholieke Universiteit Leuven, 2018.

[106] Stephen Vavasis. “On the complexity of nonnegative matrix factorization”. In: *SIAM Journal on Optimization* 20.3 (2010), pp. 1364–1377.

[107] Nico Vervliet, Otto Debals, and Lieven De Lathauwer. “Exploiting efficient representations in large-scale tensor decompositions”. In: *SIAM Journal on Scientific Computing* 41.2 (2019), A789–A815.

[108] Yu-Xiong Wang and Yu-Jin Zhang. “Nonnegative matrix factorization: A comprehensive review”. In: *IEEE Transactions on Knowledge and Data Engineering* 25.6 (2012), pp. 1336–1353.

[109] Yangyang Xu and Wotao Yin. “A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion”. In: *SIAM Journal on Imaging Sciences* 6.3 (2013), pp. 1758–1789.

[110] Yu Zhang, Guoxu Zhou, Qibin Zhao, Cichocki Andrzej, and Xingyu Wang. “Fast nonnegative tensor factorization based on accelerated proximal gradient and low-rank approximation”. In: *Neurocomputing* 198 (2016), pp. 148–154.

[111] Zhong-Yuan Zhang and Jie Zhang. “Survey on the variations and applications of nonnegative matrix factorization”. In: *Proceedings of 9th International Symposium on Operations Research and Its Applications, Chengdu-Jiuzhaigou, China*. Citeseer. 2010, pp. 317–323.

[112] Shi Zhong and Joydeep Ghosh. “Generative model-based document clustering: a comparative study”. In: *Knowledge and Information Systems* 8 (3) (2005), pp. 374–384.

[113] Guoxu Zhou, Andrzej Cichocki, Qibin Zhao, and Shengli Xie. “Nonnegative matrix and tensor factorizations: An algorithmic perspective”. In: *IEEE Signal Processing Magazine* 31.3 (2014), pp. 54–65.

[114] Guoxu Zhou, Shengli Xie, Zuyuan Yang, Jun-Mei Yang, and Zhaoshui He. “Minimum-volume-constrained nonnegative matrix factorization: Enhanced ability of learning parts”. In: *IEEE Transactions on Neural Networks* 22.10 (2011), pp. 1626–1637.

[115] Feiyun Zhu, Ying Wang, Bin Fan, Shiming Xiang, Geofeng Meng, and Chunhong Pan. “Spectral unmixing via data-guided sparsity”. In: *IEEE Transactions on Image Processing* 23.12 (2014), pp. 5412–5427.

Appendix

Symbols

§	Chapter or section symbol.
$\mathbb{R}, \mathbb{R}_+$	The set of real and nonnegative real numbers.
$\mathbb{R}^m, \mathbb{R}^m_+$	The set of real and nonnegative real m-vectors.
Δ^r	Unit simplex in $\mathbb{R}^r$.
$\mathcal{C}^r_2$	2nd-order cone in $\mathbb{R}^r$.
$\mathcal{U}^m, \mathcal{U}^m_+$	The set of unimodal and nonnegative unimodal m-vectors.
$\mathbb{R}^{m\times n}, \mathbb{R}^{m\times n}_+$	The set of real and nonnegative real $m \times n$ matrices.
$\mathbb{R}^{I\times J\times K}, \mathbb{R}^{I\times J\times K}_+$	The set of real and nonnegative real $I \times J \times K$ tensors.
$\mathbb{R}^{I_1\times\cdots\times I_N}, \mathbb{R}^{I_1\times\cdots\times I_N}_+$	The set of real and nonnegative real $I_1 \times \cdots \times I_N$ tensors.
$[a, b]$	Interval $a \leq x \leq b$.
$[n]$	Interval $[1, n]$. If n is integer, it denotes the set of integers $\{1, \ldots, n\}$.
$[\,\cdot\,]_+$	$\max\{\,\cdot\,, 0\}$, element-wise.
$\llbracket m \rrbracket$	The set of odd integers in $[m]$.
$\mathbf{v} \geq 0$, $\mathbf{M} \geq 0$, $\mathcal{T} \geq 0$	Vector $\mathbf{v}$, matrix $\mathbf{M}$, tensor $\mathcal{T}$ are elementwise nonnegative.
v_i	The ith element of vector $\mathbf{v}$.
$\|\mathbf{v}\|_2, \|\mathbf{v}\|_1, \|\mathbf{v}\|_\infty$	The l_2, l_1 and l_∞ norm of vector $\mathbf{v}$.
$\mathbf{m}_j, \mathbf{M}(:, j), \mathbf{M}_{:,j}, \mathbf{M}_{:j}$	The jth column of matrix $\mathbf{M}$.
$\mathbf{m}^i, \mathbf{M}(i, :), \mathbf{M}_{i,:}, \mathbf{M}_{i:}$	The ith row of matrix $\mathbf{M}$.
$\mathcal{T}_{[i]}$	The mode-i unfolding of tensor $\mathcal{T}$, see (6.5).
$\mathcal{T}(i, :, :), \mathcal{T}(:, j, :), \mathcal{T}(:, :, k)$	MATLAB notation of mode-1,2,3 fiber of tensor $\mathcal{T}$.
$\mathbf{A} \boxtimes \mathbf{B}$	Kronecker product of two matrices $\mathbf{A}, \mathbf{B}$, see (6.2).
$\mathbf{A} \odot \mathbf{B}$	Khatri-rao product of two matrices $\mathbf{A}, \mathbf{B}$, see (6.3).
$\mathbf{a} \otimes \mathbf{b}$	Tensor product of two vectors $\mathbf{a}, \mathbf{b}$, see (6.1).
$\langle \mathbf{a}, \mathbf{b} \rangle, \mathbf{a}^\top \mathbf{b}$	Dot product of two vectors $\mathbf{a}, \mathbf{b}$.
$M(i, j), M_{i,j}$	The entry at the ith row, jth column of matrix $\mathbf{M}$.
$\operatorname{rank}(\mathbf{M})$	The rank of matrix $\mathbf{M}$.
$\operatorname{Tr}(\mathbf{M})$	The trace of matrix $\mathbf{M}$
$\det(\mathbf{M})$	The determinant of matrix $\mathbf{M}$.
$\|\mathbf{M}\|_F, \|\mathcal{T}\|_F$	The Frobenius norm of matrix $\mathbf{M}$ and tensor $\mathcal{T}$.
$\|\mathbf{M}\|_*$	The Nuclear norm (largest singular values) of matrix $\mathbf{M}$.
$\|\mathbf{M}\|_2$	The spectral norm (largest singular value) of matrix $\mathbf{M}$.
$\sigma_i(\mathbf{M})$	The ith largest singular value of $\mathbf{M}$.
$\operatorname{cone}(\mathbf{M})$	Cone generated by the columns of the matrix $\mathbf{M}$.
$\operatorname{cone}^*(\mathbf{M})$	Dual cone generated by the columns of the matrix $\mathbf{M}$.
$\mathbf{I}_r$	Identity matrix of order r.
$\mathbf{\Pi}_r$	Permutation matrix of order r.

Glossary

AccProjG Accelerated Projected Gradient. 29, 76, 80, 97

ALS Alternating Least Squares. 113, 127

ANLS Alternating Nonnegative Least Squares. 8, 97

AO Alternating Optimization. 97

BCD Block-Coordinate Descent. 95

BSS Blind Source Separation. 56

CPD Canonical Polyadic Decomposition. 93

HALS Hierarchical Alternating Least Squares. 8, 97

HER Heuristic Extrapolation with Restarts. 100, 106

HSI Hyperspectral Imaging. 37

HU Hyperspectral Unmixing. 37

minvolNMF Minimum Volume NMF. 15

MIP Mixed-Integer Programming. 76

MM Majorization Minimization. 32, 60

MRSA Mean Removed Spectral Angle. 42

MTTKRP Matricized Tensor Times Khatri-Rao Product. 96

NMF Nonnegative Matrix Factorization. 6, 94

NNLS Nonnegative Least Squares. 7, 96

NTF Nonnegative Tensor Factorization. 94, 96

NuMF Nonnegative Unimodal Matrix Factorization. 71

QP Quadratic Programming. 31, 96

SNMF Separable NMF. 10, 39

SPA Successive Projection Algorithm. 11

SSC Sufficiently Scattered Condition. 24

SVD Singular Value Decomposition. 31