

École polytechnique de Louvain

Combining the full-space and block-coordinate approaches for exact non-negative matrix factorization

Design of conic-based 'catalyzers' towards
exactNMFs

Author: **Matthieu BOUCQUEY**
Supervisors: **François GLINEUR, Valentin LEPLAT**
Readers: **Guillaume OLIKIER, Pierre-Antoine ABSIL**
Academic year 2022–2023
Master [120] in Mathematical Engineering

Contents

Contents	i
List of acronyms	iii
Abstract	v
1 Preliminaries and background	1
1.1 NMF : An intuitive non-negative idea	1
1.1.1 Part-based, unsupervised and non-negative decomposition	1
1.1.2 Generative model and sparse factors	3
1.1.3 Interpretability and Model Identifiability	4
1.2 Problem formulations	5
1.2.1 Exact NMF	5
1.2.2 Approximate NMF	7
1.3 Geometrical interpretations	10
1.3.1 A nested polyhedral cone problem	10
1.3.2 A nested convex hull problem	11
1.4 Algorithmic approaches for approximate NMF	13
1.4.1 Taking a cue from K-means	13
1.4.2 Full-space VS block-coordinate local search	14
1.4.3 Multiplicative updates [MU]	15
1.4.4 Alternating (non-negative) least squares [ANLS, ALS]	16
1.4.5 Hierarchical alternating least squares [HALS]	18
1.4.6 Faster HALS [A-HALS, E-A-HALS]	20
1.4.7 Initialization of the factors	21
2 Existing algorithmic approaches for ExactNMF	23
2.1 BCD heuristics for ExactNMF	24
2.1.1 Multi-start heuristics [MS, MBS]	24
2.1.2 Rank-by-rank heuristic [RBR]	26
2.1.3 Simulated annealing heuristic [SA]	26
2.2 Conic-based heuristics for ExactNMF	28
2.2.1 NMO and NMU with conic optimization	28

2.2.2	<i>IPM-based Frank-Wolfe algorithm for NMO/NMU</i>	30
3	Applications of ExactNMF	35
3.1	Motivating applications for ExactNMF	35
3.1.1	<i>A non-exhaustive overview</i>	35
3.1.2	<i>Extension complexity of polyhedra</i>	36
3.2	(Open) Problems of interest	38
3.2.1	<i>Infinitesimally rigid non-negative factorizations</i>	38
3.2.2	<i>Extension complexity of n-gons</i>	38
3.2.3	<i>Linear euclidean distance matrices</i>	41
3.2.4	<i>Unique disjointness problem</i>	41
4	ExactNMF via conic-based catalyser	43
4.1	There is a place for improvement	43
4.1.1	<i>"Brutality" of the SPI refinement</i>	43
4.1.2	<i>Towards a smooth and cautious refinement</i>	44
4.2	Change of paradigm for conic NMO/NMU	45
4.2.1	<i>Limit the use of IPMs in CFW</i>	46
4.2.2	<i>Convergence-aware conic-BCD switches</i>	47
4.2.3	<i>Switching back to CFW?</i>	48
4.3	Conic-based heuristics for ExactNMF	49
4.3.1	<i>"Fully initialized" heuristics $[C, C_{pSPI}, C_{TP}, C_{TMBS}]$</i>	49
4.3.2	<i>"Partially initialized" heuristics $[C_{RBR}, C_{TSA}, C_{HYB}]$</i>	52
4.3.3	<i>Simultaneity : Necessary condition and great asset</i>	54
4.3.4	<i>Initialization of the conic-based heuristics</i>	56
5	Numerical Experiments	57
5.1	Towards scalability : Unlock higher dimensional problems	57
5.1.1	<i>Construction of the conic optimization problem</i>	57
5.1.2	<i>Making LMO iterations cheaper?</i>	57
5.2	Numerical validation experiments	60
5.2.1	<i>Nested hexagons</i>	60
5.2.2	<i>Infinitesimally rigid non-negative factorizations</i>	61
5.2.3	<i>Mid-term review</i>	63
5.3	Main numerical experiments	63
5.3.1	<i>Generic n-gons</i>	63
5.3.2	<i>Random linear Euclidean distance matrices</i>	65
5.3.3	<i>Open question on a UDISJ matrix</i>	67
6	Discussion and Conclusion	69
	Bibliography	71

List of acronyms

A-HALS	Accelerated hierarchical alternating least square algorithm.
ALS	Alternating least square algorithm.
ANLS	Alternating non-negative least square.
BCD	Block-coordinate descent.
C_HYB	Hybrid conic-based heuristic.
C_RBR	Rank-by-rank conic-based heuristic.
C_TMBS	Threshold multi-best-start conic-based heuristic.
C_TSA	Threshold simulated annealing conic-based heuristic.
CFW	Frank-Wolfe algorithm with a conic LMO .
HALS	Hierarchical alternating least square algorithm.
HYB	Hybrid BCD-based heuristic.
IPM	Interior-point method.
FW	Frank-Wolfe algorithm.
LDR	Linear dimensionality reduction.
LRMA	Low rank matrix approximation.
MBS	Multi-best-start BCD-based heuristic.
MS	Multi-start BCD-based heuristic.
MU	Multiplicative updates algorithm.
NMF	Non-negative matrix factorization.
NMO	Non-negative matrix over-approximation.
NMU	Non-negative matrix under-approximation.
RBR	Rank-by-rank BCD-based heuristic.
RCB	Rectangle covering bound.
RRCB	Refined rectangle covering bound.
SA	Simulated annealing BCD-based heuristic.
SPI	Singular pattern integration.
SVD	Singular value decomposition.

Abstract

Nonnegative Matrix Factorization (NMF) is a widespread linear dimensionality reduction technique which decomposes a nonnegative matrix X as the product of two low-rank nonnegative matrices W and H of appropriate dimensions.

In the linear algebra literature, NMF has mostly been studied under its approximate form, i.e. $X \approx WH$. However, its exact formulation $X = WH$ and the closely related concept of *non-negative rank* are far from irrelevant. In fact, they arise in a wide range of domains including combinatorial optimization [Yan88], communication complexity [Lov89], complexity theory [Nis91], quantum mechanics [SS93], computational geometry [GG12b], probability theory [CR10; PA71] or even graph theory [Fio+13].

Finding exactNMFs, i.e. factors W and H such that $X = WH$ for a certain fixed factorization rank r , has proved to be a difficult problem, even for a small factorization rank r [Vav10; Aro+12]. In the absence of applicable theoretical results, the development of effective computational techniques to find exactNMFs has appeared as an important direction of research.

Different exploration strategies for the highly non-convex solution space of the NMF problem have been proposed. First, some algorithmic approaches relying on state-of-the-art *block-coordinate descent* (BCD) methods designed to address the approximate NMF problem have been introduced [Van+16]. Secondly, other methods taking root in a conic optimization reformulation of the NMF problem have been imagined [Lep+21]. The latter has opened the door to an interesting full-space exploration of the NMF solution space but relies on poorly scalable *interior-point methods* (IPM). In both cases, heuristics look for high-quality solutions hoping to find exactNMFs.

In a wide variety of settings, BCD-based heuristics perform sufficiently well. However, in some particularly difficult cases, they seem to seriously struggle to find the desired exactNMFs. On this basis, this thesis explores the potential of using the conic reformulations of NMF introduced in [Lep+21] as catalyzers to perform a better solution space exploration towards exactNMFs while limiting the computational weight of the IPMs as much as possible.

The heuristics proposed in this thesis are an attempt to reconcile, in an optimized way, the good properties of both paradigms: the fast full-space mobility of the conic-based approach and the iteration speed and the good convergence properties of the BCD

approach. The final goal is to compute exactNMFs where the existing heuristics cannot (whatever the computational load) and thereby complement the successful heuristics and local search algorithms that have already been proposed.

Using well-chosen numerical experiments, we have shown that conic-based heuristics introduced in this thesis constitute (at least) a successful complement to the state-of-the-art heuristics and local search algorithms that have already been proposed in the NMF context. In fact, for many matrices for which the BCD-based methods have proved incapable of finding exact factorizations, the conic-based heuristics have reached better factorizations and even found exactNMFs.

In Chapter 1, we recall the fundamentals of NMF: intuition, problem formulations, properties, geometrical interpretations, and algorithms. Chapter 2 presents some motivating applications of the exact NMF formulation with a special focus on specific settings in which we will test the new conic-based heuristics that are introduced in Chapter 3. Finally, in Chapter 4, we report the concrete numerical experiments that have been conducted to evaluate their performance.

Preliminaries and background

NON-NEGATIVE Matrix Factorization (NMF) is a widespread and easily interpretable dimensionality reduction technique which decomposes any non-negative matrix $\mathbf{X}$ as the product of two low-rank non-negative matrices $\mathbf{W}$ and $\mathbf{H}$ of appropriate dimensions.

The NMF paradigm was introduced by Paatero and Tapper in 1994 [PT94] and has gained momentum with the seminal article by Lee and Seung [LS99]. The latter conceptually revealed the full potential of NMF as a new powerful tool for the analysis of non-negative data. Since then, NMF and its applications have been a very active field of research, see Figure 1.1.

1.1 NMF : An intuitive non-negative idea

Let us first identify the underlying concepts and assumptions behind NMF and thereby understand its place within the huge family of data analysis techniques.

1.1.1 Part-based, unsupervised and non-negative decomposition

As exposed in [Ngo08], the very natural idea behind NMF is that

"Any object is made up of a certain number of elementary parts."

As humans, we assume this constructive principle to either build or recognize objects. This assumption suggests that each object O is the product of a receipt looking like

$$O = \alpha_1 \cdot E_1 + \alpha_2 \cdot E_2 + \dots + \alpha_i \cdot E_i + \dots \quad (1.1)$$

where E_i is the elementary part i and α_i its quantity. In the context of algebraic objects, this principle is better known as *linear dimensionality reduction* (LDR). Mathematically,

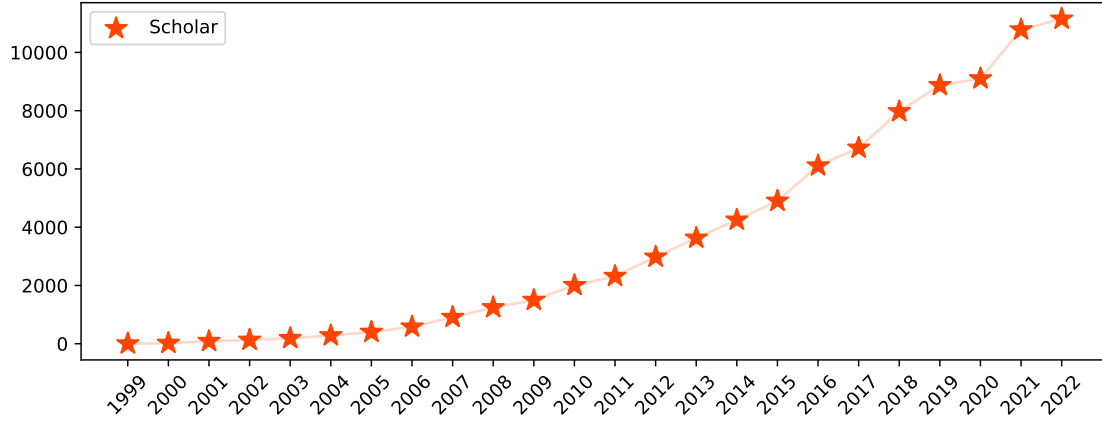


Figure 1.1 Number of search results for papers containing either ‘non-negative matrix factorization’ or ‘nonnegative matrix factorization’ on Google Scholar.

given a data set of n data points $x_j \in \mathbb{R}^m$, LDR looks for a small number $r \ll n$ of basis vectors $w_k \in \mathbb{R}^m$ such that each data point is well-approximated by a linear combination of the basis vectors, i.e.

$$x_j \approx \sum_{k=1}^r w_k h_{kj} \quad (1.2)$$

Considering all data points side-by-side in a concatenated matrix format, it becomes clear that LDR is equivalent to *low-rank matrix approximation* (LRMA), i.e.

$$[x_1, x_2, \dots, x_n] \approx [w_1, w_2, \dots, w_r] \cdot [h_1, h_2, \dots, h_n] \quad (1.3)$$

where $h_i \in \mathbb{R}^r$ are the reconstruction weights to be applied to the basis vectors to approximate x_i . Following the notation chosen in Gillis’ reference handbook [Gil20], in matrix form, this gives

$$X \approx W \cdot H \quad (1.4)$$

Of course, the quality of this approximation must be assessed by an appropriate *distance measure* which has to be selected according to the noise statistic assumed on the data. The most widely used measure is the Frobenius norm of the error.

$$\|X - WH\|_F \quad (1.5)$$

It implicitly assumes that the data matrix X is contaminated with a Gaussian noise N (such that $X = WH + N$) which is reasonable in many practical situations [Gil20].

Over the years, LRMA methods have gathered more and more interest and have been intensively developed to compress data, filter noise, reduce the computational effort for further manipulation of the data, or directly identify the hidden structure in a data set.

Different constraints can be imposed on the factors W and H . These constraints depend on the application at hand and allow, for example, for meaningful interpretation of the factors. As its name suggests, in the case of NMF, we assume elementwise non-negativity for both the data matrix X and the low-rank factors W and H making NMF an instance of a *constrained low-rank matrix approximation* (CLRMA) problem with

$$W \geq 0 \quad H \geq 0 \quad (1.6)$$

This non-negative framework is particularly relevant since in a huge variety of physical settings, we encounter non-negative quantities, e.g., image pixels, concentrations of substances, absolute temperatures, and probabilities. As suggested in [LS99], imposing non-negativity on the factors preserves the possible physical interpretation of the extracted 'parts'. NMF draws its relevance in the fact that standard tools such as *Singular Value Decomposition* (SVD) fail in providing this meaningful insight into the data. Furthermore, other analysis methods like *Principal Component Analysis* (PCA) require a certain type (e.g., zero-sum, orthogonal) of extracted feature which non-negative objects by their nature never or hardly possess. Unfortunately, asking for high-quality features has a cost. The complexity of the NMF problem will be discussed further in section 1.2.1.

It is here also worth mentioning that NMF belongs to the *unsupervised learning* family as it performs an 'automatic' and complete extraction of meaningful patterns from a given non-negative data structure.

1.1.2 Generative model and sparse factors

From (1.2), it is clear that NMF adopts an additive structure. But the non-negativity constraint on the basis vectors leads to an even more interesting propriety, namely 'non-subtractivity', i.e. the absence of subtractions in the model. By intuition, we understand that this excludes any possibility of compensations and therefore significantly reduces the degrees of freedom within the decomposition model which will increase the complexity of the problem (see section 1.2.2). All in all, NMF assumes a generative model for the data hence "*learning the parts*" in [LS99].

In addition, this strict additivity machinery also leads the parts or basis elements to be sparse. More formally, the reason for this behaviour is that stationary points $(W; H)$ of the NMF problem will be typically located at the boundary of the feasible domain $\mathbb{R}_+^{m \times r} \times \mathbb{R}_+^{r \times n}$, hence will feature zero components [GG10]. This can be explained with the First-order optimality conditions. In fact, the set of stationary points of a problem of the type

$$\min_{\mathbf{x} \in \mathbb{R}^N} f(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{x} \geq 0 \quad (1.7)$$

is given by the following expression (where $\nabla f(\mathbf{x})$ is the gradient of f)

$$S_f = \{\mathbf{x} \in \mathbb{R}^N | \mathbf{x} \geq 0, \nabla f(\mathbf{x}) \geq 0 \text{ and } x_i [\nabla f(\mathbf{x})]_i = 0 \forall i\} \quad (1.8)$$

where we expect some components of the solution to be equal to zero.

In addition to the reduction in memory requirements for the basis vectors, sparsity improves the interpretation of the factors (especially in clustering contexts) and therefore appears as a real competitive advantage of NMF over the other data analysis alternatives.

1.1.3 Interpretability and Model Identifiability

As we mentioned before, it is really the interpretability argument put forward in [LS99] that brought NMF to the forefront. A huge number of interpretable NMF results have been obtained since the early 2000's underscoring, even more, this important asset [Fu+19]. Since the final goal is to recover the unique ground-truth generative factors, it seems very natural to relate the interpretability of NMF to its model uniqueness, or, latent factor identifiability.

At first glance, we could argue that NMF irrevocably implies non-uniqueness [Lau+08]. As a proof, even if one is given an optimal NMF (W^*, H^*) of X , there might exist many equivalent solutions $(W^*Q, Q^{-1}H^*)$ for non-monomial¹ matrices Q with $UQ \geq 0$ and $Q^{-1}V \geq 0$. However, as indicated in [Fu+19], when studying identifiability, we often ignore cases where Q only results in column reordering and scaling/counter-scaling of W and H , which is considered unavoidable, but also inconsequential in most applications. Bearing this in mind, let us formally define the identifiability of nonnegative matrix factorization:

Definition 1.1 (Identifiability). Consider a data matrix generated from the model $X = W_{\#}H_{\#}$, where $W_{\#}$ and $H_{\#}$ are the ground-truth factors. Suppose that $W_{\#}$ and $H_{\#}$ satisfy a certain condition. Let (W^*, H^*) be an optimal solution to an identification criterion or an output from a procedure. If, under the aforementioned condition of $W_{\#}$ and $H_{\#}$'s, it holds that

$$W^* = W_{\#}\Pi D \quad H^* = D^{-1}\Pi^{\top}H_{\#} \quad (1.9)$$

where Π is a permutation matrix and D is a full-rank diagonal matrix, then we say that the matrix factorization model is identifiable under that condition.

With this in mind, if we assume a situation where data has true generative elements, it seems legitimate to ask the following question: When does non-negative matrix factorization give a correct decomposition into parts? [DS03]

¹A monomial matrix is a permutation of a diagonal matrix with positive diagonal elements. The inverse of a non-negative matrix is usually not non-negative. Non-negative monomial matrices constitute an exception: a non-negative matrix has a non-negative inverse if and only if it is a (non-negative) monomial matrix. See [DeM72] and references therein.

Intuitively, adding constraints such as nonnegativity on the latent factors may help us achieve identifiability since constraints can reduce the ‘volume’ of the solution space and remove a lot of possibilities for Q . Unfortunately, for many non-negative matrices, this is not enough to ensure their non-negative factorization uniqueness. Striving for even further improvement, linear algebra researchers have come up with many variants of the NMF problem to overcome this nonuniqueness problem [Gil12]. The (more) well-posed NMF problems tend to use regularization in order to obtain solutions with some structure (e.g. sparsity, volume) or additional constraints (e.g. orthogonality, symmetry) to restrict even more the solution space. See [Gil20] for a more complete overview of the existing variant formulations.

1.2 Problem formulations

When we refer to NMF, depending on the application, we are implicitly pointing to either the *exact* or the *approximate* version of the same non-negative factorization problem. It is key to formalize both formulations in detail since this thesis will mainly focus on the interaction between these two formulations and since each formulation has its own theoretical and applicative interests.

1.2.1 Exact NMF

Like other matrix factorizations, the NMF first arose as an exact factorization problem defined as follows.

Problem 1.2 (Fixed_ExactNMF). Given a non-negative matrix $X \in \mathbb{R}_+^{m \times n}$ and a factorization rank r , compute, if possible, two non-negative matrices $W \in \mathbb{R}_+^{m \times r}$ and $H \in \mathbb{R}_+^{r \times n}$ such that $X = WH$. We refer to WH as an *exactNMF* of X of size r .

Fixing the factorization rank r is a very strong assumption. Relaxing this constraint gives an even more interesting but more difficult problem to solve.

Problem 1.3 (ExactNMF). Given a non-negative matrix $X \in \mathbb{R}_+^{m \times n}$, compute the two smallest non-negative matrices $W \in \mathbb{R}_+^{m \times r}$ and $H \in \mathbb{R}_+^{r \times n}$ such that

$$X = WH. \tag{1.10}$$

The latter *ExactNMF* problem is closely related to the quantity referred to as the *non-negative rank* of X which is the smallest r such that X admits an exactNMF of size r , and is denoted $\text{rank}_+(X)$ [CR93].

Problem 1.4 (NN_Rank). Given a non-negative matrix $X \in \mathbb{R}_+^{m \times n}$, find the minimum value of r such that $X = WH$, $W \in \mathbb{R}_+^{m \times r}$, $H \in \mathbb{R}_+^{r \times n}$ and W, H have non-negative entries.

It is here worth mentioning that the non-negative rank depends on the underlying field [Shi15]. An irrationality phenomenon was brought to light by providing rational matrices with different non-negative ranks over $\mathbb{R}$ and $\mathbb{Q}$ [Shi16; Chi+17]. In this thesis, we will work exclusively on the reals.

Unfavourable complexity...

Over the years, no algorithm in the literature came up with a polynomial-time guarantee to solve **ExactNMF**. Ultimately, Vavasis [Vav10] proved the hypothesis that this problem is difficult to solve. He showed that checking whether

$$\text{Rank}(X) \stackrel{?}{=} \text{Rank}_+(X) \quad (1.11)$$

is NP-hard². Since determining the non-negative rank is a generalization of (1.11), the results in [Vav10] imply that solving **NN_Rank** and by extension **ExactNMF** are also NP-hard.

...How should we approach ExactNMF ?

Failing to find the answer to the famous question $P \stackrel{?}{=} NP$ [Coo00], the scientific community has worked hard to find alternative ways to address the **ExactNMF** problem. From a theoretical point of view, Arora et al. [Aro+12] showed that checking whether a non-negative matrix admits a rank- r exactNMF, i.e. **Fixed_ExactNMF**, can be done in time polynomial³ in m and n (i.e., considering the factorization rank r fixed). It is proven that by a clever reformulation of the **Fixed_ExactNMF** problem for a $m \times n$ matrix as a system of $\mathcal{O}(mn)$ fixed-degree polynomial equalities involving $\mathcal{O}(r^2 2^r)$ variables, it can be solved with total complexity $\mathcal{O}((mn)^{r^2 2^r})$. Moitra later improved this complexity result to $\mathcal{O}((mn)^{r^2})$ in [Moi16].

Although these research efforts have resulted in promising theoretical guarantees, unfortunately, they do not directly translate into practical algorithms, even when dealing with very small matrices. Even with modest m and n the power involving r makes the problem out of a reasonable calculation range. In the face of this persistent lack of applicable results, the development of effective computational techniques to solve the **Fixed_ExactNMF** problem for small matrices appeared as an important direction of research. In this vein, approaches relying on state-of-the-art methods to solve the

²This was proved by showing that **ExactNMF** is equivalent with a problem in polyhedral combinatorics called *intermediate simplex*.

³One might think that there is a contradiction between Vavasis pretending that **ExactNMF** NP-hard and Arora saying that it is polynomial. But, in fact, Arora assumes that the factorization rank r is fixed (and it intervenes in the exponent of the polynomial), whereas Vavasis lets the rank r grow with the dimensions of the matrix.

approximate NMF problem with high-quality solutions were developed with relative success, see section 2.1.

ExactNMF via bounds on NN_Rank

Another axis of research focuses on developing lower and upper bounds for the non-negative rank. These theoretical guarantees are of utmost importance since they are generally computationally more efficient⁴ and enable us to narrow down the range of possible values for $\text{rank}_+(X)$ and therefore prevent ‘exactNMF seekers’ from spending useless computational effort on wrong values.

First, it is interesting to note that upper bounds on the non-negative rank can be obtained from any factorization of the given non-negative matrix. Explicit theoretical bounds have still been successfully developed [Shi14b]. On the other hand, determining nontrivial lower bounds on the non-negative rank is in general more challenging. A larger variety of approaches were used to obtain them. They can be classified into three classes [VGG17].

Geometrical bounds. Some bounds were obtained from a geometric representation of the matrix, i.e, representing the matrix as a polytope or a pair of polytopes and bounding the nonnegative rank based on characteristics of this representation [DGG21; GG12b].

Combinatorial bounds. Others are based on the sparsity pattern of the input matrix. The most well-known one is the *rectangle covering bound* (RCB) that counts the minimum number of rectangles necessary to cover all positive entries of the matrix, a rectangle being a subset of rows and columns for which the corresponding submatrix contains only positive entries [Fio+13]. An even better bound is the *refined rectangle covering bound* (RRCB) introduced by Oelze, Vandaele, Weltge [OVW14]. Although these bounds can be rather strong in some cases, they are computationally very expensive and only work well for matrices with ‘well located’ zero entries.

Convex relaxation bounds. Finally, unlike many of the existing bounds, Fawzi and Parrilo found lower bounds that do not solely depend on the sparsity pattern of the matrix and are applicable to matrices with arbitrary support including the ones with strictly positive entries. Formally, they are expressed as the solution of a copositive programming problem and can be relaxed to obtain polynomial-time computable lower bounds using semidefinite programming (see [FP16] and [FP15]).

1.2.2 Approximate NMF

For most machine learning and data mining applications, it typically does not make sense to look for exactNMFs because the data is usually contaminated with noise. In such settings, the main goal is to recover the underlying generative elements and their associated reconstruction coefficients. This is achieved by minimizing an appropriate

⁴For example, the rank is a lower bound on the non-negative rank and there are polynomial algorithms to compute the rank of a matrix.

distance measure $D(X, WH)$ which has to be selected according to the noise statistic assumed on the data. For a complete overview, we refer to section 5.1 in [Gil20] and the references therein. In this thesis, we restrict ourselves to the $\mathbf{l}_2$ -NMF problem which is based on the Frobenius norm which is the most standard setting.

Problem 1.5 ($\mathbf{l}_2$ -NMF). Given a non-negative matrix $X \in \mathbb{R}_+^{m \times n}$ and a factorization rank r , compute two non-negative matrices $W \in \mathbb{R}_+^{m \times r}$ and $H \in \mathbb{R}_+^{r \times n}$ such that the elementwise squared reconstruction error is minimized, i.e.

$$(W^*, H^*) = \arg \min_{W \in \mathbb{R}_+^{m \times r}, H \in \mathbb{R}_+^{r \times n}} \|X - WH\|_F^2 \quad (1.12)$$

From a ‘rank-by-rank’ view point, this is equivalent to

$$(d_k, s_k^*, t_k^*)_{k=1}^r = \arg \min_{\substack{s_k \in \mathbb{R}_+^m, \|s_k\|=1 \\ t_k \in \mathbb{R}_+^n, \|t_k\|=1 \\ d_k \in \mathbb{R}_+}} \left\| X - \sum_{k=1}^r d_k \cdot s_k t_k^\top \right\|_F^2 \quad (1.13)$$

where d_i ’s indicate the relative importance of each rank-1 non-negative contribution.

Constructivity

Building on the ‘rank-by-rank’ perspective, it is interesting to note that its less restricted counterpart, namely the SVD, can boast a powerful asset: the *Eckart-Young-Mirsky theorem*. As a reminder,

Theorem 1.6 (Eckart–Young–Mirsky). Given any matrix $X \in \mathbb{R}^{m \times n}$, its truncated SVD of rank r , i.e.

$$\tilde{X}_r = \sum_{k=1}^r \sigma_k u_k v_k^\top \quad (1.14)$$

is its best rank r approximation. Formally,

$$\tilde{X}_r = \arg \min_{\text{rank}(M) \leq r} \|X - M\|_F^2 \quad (1.15)$$

Since this theorem holds for any rank $1 \leq r \leq \min(n, m)$, it provides the SVD with a ‘monotonic constructivity’ property within the LRMA framework. Unfortunately, with its additional non-negative restrictions, NMF does not benefit from this key property which closes the door to very straightforward progressive/rank-by-rank algorithmic approaches.

Nevertheless, we will see in section 2.1.2 that in practice rank-by-rank strategies can still be effective since some rank tendencies are maintained along the optimization process.

Optimality issues

As might be suggested by the [Eckart–Young–Mirsky](#) theorem, the unconstrained low-rank matrix approximation in the Frobenius norm (i.e. (1.12) without the non-negativity constraints) is tractable. It is a non-convex⁵ optimization problem but, surprisingly, it has a very nice property: every local minimum is global, and all other stationary points are saddle points [[Zhu+18](#)]. Note that the unconstrained least-squares error problem can be reformulated as a *semidefinite program* (SDP) by using the Ky Fan 2-k-Norm [[DV16](#), Prop. 2.9].

Unfortunately, the additional non-negativity requirements composing $\mathbf{l}_2$ -NMF destroy this nice geometrical and make the problem NP-hard in general⁶. This can be explained by the fact that the global minimum (W^*, H^*) guaranteed by the [Eckart–Young–Mirsky](#) theorem is in general not located within the non-negative orthant. And the very bad news is that non-negativity constraints tend to generate numerous spurious local minima and saddle points on the boundary of the non-negative orthant [[Gil20](#)].

It is here very important to highlight the fact that, since the problem highly differs from one matrix to the other, the ‘level of non-convexity’ and more generally the structure of the solution space will greatly vary making the problem more or less difficult to solve with local iterative optimization schemes. The shape and location of the basins of attraction, local minima, and plateaus can lead to potential difficulties to perform a successful exploration of the search space towards a global minimum.

All the previous observations regarding the $\mathbf{l}_2$ -NMF problem clearly converges towards the following conclusion :

“The better interpretability provided by the NMF over its competitors is obtained at the cost of an increase in the complexity of the factorization problem and by extension in the difficulty of solving it.”

Unfortunately, there does not exist a successful convexification approach for NMF, as opposed to other low-rank models [[Gil17](#)]. It is here worth mentioning that a convexification approach to compute lower bounds for the non-negative rank has already been proposed [[FP15](#)] (see section 1.2.1).

Rank-one solution of $\mathbf{l}_2$ -NMF is tractable

While the problem is difficult in general, finding the best rank 1 non-negative approximation is within reach [[Gil20](#)].

Theorem 1.7. $\mathbf{l}_2$ -NMF with $r = 1$ can be solved in polynomial time.

⁵The objective function of (1.12) is composed of squared differences including products of the entries of the two factors W and H .

⁶This follows from the NP-hardness result of Vavasis [[Vav10](#)] for [ExactNMF](#) (see section 1.2.1). When there exists an exactNMF of size r such that $X = WH$, solving $\mathbf{l}_2$ -NMF allows us to recover it when its objective function satisfies $\|X - WH\|_F = 0$. Therefore, any hardness result that applies to [ExactNMF](#) applies to $\mathbf{l}_2$ -NMF. To sum up, “ $\mathbf{l}_2$ -NMF is at least as hard as [ExactNMF](#)”.

1 | Preliminaries and background

Proof. Let $\sigma_1 u_1 v_1^\top$ be the rank-one truncated SVD of X , which can be computed in polynomial time (see [GV13b]). Since we have

$$\|X - u_1 v_1^\top\|_F^2 \geq \|X - |u_1 v_1^\top|\|_F^2 = \|X - |u_1| |v_1|^\top\|_F^2 \quad (1.16)$$

where $|M|$ denotes the componentwise absolute value of matrix M .

It follows by [Eckart–Young–Mirsky](#) theorem that

$$W = \sqrt{\sigma_1} |u_1| \quad H = \sqrt{\sigma_1} |v_1| \quad (1.17)$$

is an optimal solution of $\mathbf{l}_2$ -NMF with $r = 1$. □

The latter theorem is closely related to another key theorem about non-negative matrices: *The Perron-Frobenius theorem*.

Theorem 1.8 (Perron-Frobenius, see [BP94]). Let X be a square⁷ non-negative matrix. There exists a largest modulus eigenvalue of X which is non-negative and a non-negative eigenvector, called *Perron vector* corresponding to it.

The rank-1 optimal solution will be used in algorithmic rank-by-rank approaches (see sections 2.1.2 and 4.3.2).

1.3 Geometrical interpretations

To properly approach a problem and understand intuitively and in depth the issues it contains, it is rarely superfluous to dwell on the geometrical interpretation of it.

To start with, it is interesting to note that before Lee and Seung definitely set the ball rolling with their article in the prestigious *Nature* journal [LS99], they had already introduced a fertile geometrical soil to grow NMF. In fact, in [LS96], they presented a general framework for unsupervised learning by means of conic encoders which already established one of the possible geometrical intuitions behind NMF.

1.3.1 A nested polyhedral cone problem

As emphasized in section 1.1.1, the exact rank- r factorization $X = WH$ (1.10) says that there are r ‘basis’ vectors $w_i \in \mathbb{R}^m$ such that all the data points x_i have a representation as non-negative linear combinations of these vectors. This algebraic characterization has a geometric counterpart in terms of polyhedral cones [DS03] which can be defined as follows.

⁷For a rectangular non-negative matrix, similar results can be established for the largest singular value and its corresponding singular vectors.

Definition 1.9. A cone⁸ K having a finite number n of extremal rays is called *polyhedral cone*.

The factorization (1.10) tells us geometrically that the data points x_i all lie in $\text{cone}(W)$, the polyhedral cone generated by the r basis vectors w_i . In other words, all the columns of X are conic combinations of the columns of W , i.e. linear combinations with non-negative weights. The dimension of $\text{cone}(W)$ is the dimension of the subspace spanned by X in the positive orthant and is equal to $\text{rank}_+(X)$. Therefore we can geometrically rephrase the **ExactNMF** problem in terms of cones as follows.

Problem 1.10 ("Conic" **ExactNMF**). Given n non-negative matrix data points $x_i \in \mathbb{R}_+^m$, find the polyhedral cone $\text{cone}(W)$ of smallest dimension r containing all data points,

$$\text{cone}(X) \subseteq \text{cone}(W) \subseteq \mathbb{R}_+^m \quad (1.18)$$

i.e. all x_i 's can be expressed as conic combinations of the r generating extreme rays w_i .

ExactNMF is here reformulated as a *nested cone problem*, i.e. find the smallest amount r of vectors within the non-negative orthant whose conical hull contains the cone generated by the data points.

1.3.2 A nested convex hull problem

The geometric interpretation of the **ExactNMF** problem in terms of nested cones can be reformulated into an equivalent interpretation using nested convex hulls, that is, nested polytopes [Gil20]. The latter is often preferred since the problem of nested convex hulls has a long history in computational geometry and it leads to easier visualization.

Let us first introduce the following two assumptions that can be made without loss of generality [Gil17]:

- The matrices X and W do not contain columns equal to the zero vector. If this is the case, they can simply be removed as they are superfluous in the factorization.
- The columns of X and W have l_1 norm equal to one, i.e.

$$\|X(:,i)\|_1 = \|W(:,k)\|_1 \quad \forall i,k \quad (1.19)$$

Following the notation used in [Gil17], this l_1 -normalization assumption can be obtained from any non-negative factorization by the following operation.

$$XD_X^{-1} = WD_W^{-1}D_WHD_X \quad (1.20)$$

⁸A more in-depth look at cones and more specifically polyhedral cones can be found in [HS20].

1 | Preliminaries and background

where D_M is a diagonal matrix such that $D_M(j, j) = \|M(:, j)\|_1$. This MD_M^{-1} operation is referred to as the *pullback map* in [CL08].

The l_1 -normalization assumption leads to the following key theorem.

Theorem 1.11. [CR93, Theorem 3.2] Let (W, H) be any factorization of X . If the entries in each column of X and W sum to one, then the entries in each column of H sum to one.

Proof. This can be directly obtained through the following equalities where e is the vector of all ones of appropriate dimension.

$$e^\top = e^\top X = e^\top WH = e^\top H \quad (1.21)$$

Hence $e^\top = e^\top H$, that is, the entries in each column of H sum to one. $\square$

By denoting $\theta(M)$ the column- l_1 -normalized version of matrix M , since from theorem 1.11 we know that the normalization assumption implies that H is column-stochastic, it become clear that the columns of X are expressed as convex combinations of columns of W and that as a result

$$\begin{aligned} \text{cone}(X) &\subseteq \text{cone}(W) \subseteq \mathbb{R}_+^m \\ &\Updownarrow \\ \text{conv}(\theta(X)) &\subseteq \text{conv}(\theta(W)) \subseteq \Delta^m \end{aligned} \quad (1.22)$$

where $\text{conv}(M)$ is convex hull of the columns of the matrix M , i.e.

$$\text{conv}(M) = \left\{ x \mid x = My, y \geq 0, e^\top y = 1 \right\} \quad (1.23)$$

and Δ^m the unit simplex of dimension m , i.e. $\Delta^m = \text{conv}(I_m)$ with I_m being the $m \times m$ identity matrix.

Considering previous geometrical developments and notations, we can rephrase the **ExactNMF** problem as a *nested convex hull problem*.

Problem 1.12 ("Convex" **ExactNMF**). Given n non-negative matrix data points $x_i \in \mathbb{R}_+^m$, find the convex hull $\text{conv}(\theta(W))$ of smallest dimension r containing all the l_1 -normalized data points,

$$\text{conv}(\theta(X)) \subseteq \text{conv}(\theta(W)) \subseteq \Delta^m \quad (1.24)$$

i.e. all $\theta(x_i)$'s can be expressed as convex combinations of the r generating extreme rays $\theta(w_i)$.

Both the conical and convex viewpoints provide us with a geometrical insight into NMF and its properties. In particular, the non-uniqueness can better be understood.

1.4 Algorithmic approaches for approximate NMF

As we discovered in section 1.2.2, l_2 -NMF is a difficult problem to solve. However, in this section, we will see that iterative methods with low computational load iterations are still possible. Most of the proposed iterative methods have computational load per iteration roughly equivalent to the one of SVD, i.e. mnr . As suggested in [Ngo08], the main aspects that differentiate these methods are then: to which solution they tend to converge? how fast do they converge? and how to drive them to converge to solutions that possess some desired properties?

This section aims at introducing some key algorithmic approaches that have been proposed to tackle the approximate NMF problem. The best-performing among them will be needed to build new heuristics in the next sections.

1.4.1 Taking a cue from K-means

To introduce the dominating framework to deal with the NMF problem let us observe that it is a generalization of the well-known *K-Means method* [Mac67] from 1967, applied to non-negative data [Ngo08]. Suppose we have n non-negative data points $x_1, \dots, x_n$ in $\mathbb{R}^m$ and r initial centroids $w_1, \dots, w_r$ representing r clusters $C_1, \dots, C_r$. The K-Means method repeats the following two steps until convergence:

1. For each x_i , assign it to C_j if w_j is the nearest centroid to x_i , with respect to Euclidean distance.
2. For each w_j , replace it with the arithmetic mean of all x_i in C_j .

By constructing a matrix W with w_j as its columns and by creating a matrix H such that

$$H(i, j) = \begin{cases} 1 & \text{If } x_i \in C_j \\ 0 & \text{Otherwise} \end{cases} \quad (1.25)$$

it becomes clear that, if we concatenate all the data vectors in a matrix X , the K-Means method can formally be expressed as follows.

Problem 1.13 (K_Means). Given any matrix $X \in \mathbb{R}^{m \times n}$, the optimal r clusters w.r.t. the Euclidean distance are given by the columns of W^* obtained by solving the following restricted least square optimization problem:

$$(W^*, H^*) = \arg \min_{\substack{W \in \mathbb{R}_+^{m \times r} \\ H \in \{0,1\}^{r \times n}}} \|X - WH\|_F^2. \quad (1.26)$$

The assignments to the clusters are specified by the rows of H^* .

From the previous developments, it can be seen that $\mathbf{l}_2\text{-NMF}$ is different from $\mathbf{K_Means}$ only in the restrictions required from coefficient-matrix H .

Using the latter framework, it can easily be understood that the two above iterative steps amount to solve optimally the two following sub-problems at iteration $k = 0, 1, \dots$:

$$\begin{aligned} \text{[P1]} \quad H_{k+1} &= \arg \min_{H \in \mathbb{B}^{r \times n}} \|X - W_k H\|_F^2 \\ \text{[P2]} \quad W_{k+1} &= \arg \min_{W \in \mathbb{R}_+^{m \times r}} \|X - W H_{k+1}\|_F^2 \end{aligned} \quad (1.27)$$

where $\mathbb{B}^{r \times n}$ is the set of $r \times n$ binary matrices in which each column contains one and only one element equal to 1. This iterative method clearly bypasses the non-convexity issue of the problem by adopting an *alternating block-coordinate* optimization scheme which will be extensively readapted for NMF.

1.4.2 Full-space VS block-coordinate local search

As highlighted previously in this thesis, the $\mathbf{l}_2\text{-NMF}$ is (highly) non-convex. Section 1.4.1 clearly shows how tempting and natural it is to recover in some way the nice convexity properties. In this regard, in the NMF context, it seems appropriate to classify algorithms into two categories according to the search space: full-space search and (block-)coordinate search [HVB11].

Full-space search. Algorithms in the former category try to find updates for both W and H at the same time. This requires coming up with a descent direction in the $(m + n)r$ -dimensional space. Note that the simultaneity of the updates makes it easier to deduce associated convergence results.

Block-Coordinate search. Algorithms in the latter category, on the other hand, find updates for each block-coordinate in order to guarantee the descent of the objective function. Usually, search subspaces are chosen to make the objective function convex so that efficient methods can be applied. This *divide and conquer* approach might lead to the loss of some convergence properties.

Many partitions of the variables that preserve convexity of the alternating minimization sub-problems [GG08a] are possible. Since the cost function of $\mathbf{l}_2\text{-NMF}$ can be expressed as $\|X - \sum_{k=1}^r W(:,k)H(k,:)\|_F^2$ it becomes clear that convexity is only lost as soon as one column of W and the corresponding row of H are optimized simultaneously. The corresponding minimization sub-problem can, in that case, no longer be efficiently solved up to global optimality.

It is important to highlight the fact that, in the absence of a global optimization method, all the methods presented in this thesis belong to the **local search** family, i.e. they iteratively move from solution to solution in the search space by applying local changes,

until a solution deemed optimal is found or a time-bound is elapsed [Wik22b]. In most cases, the best guarantee provided by the methods is to converge to stationary points (so not even necessarily local minima).

1.4.3 Multiplicative updates [MU]

Before taking a closer look at the most successful alternating optimization schemes, let us first recall the original algorithmic idea of Lee and Seung [LS99; LS00]: *Multiplicative Updates* (MU). The MU algorithms consist of simple and simultaneous updates for W and H that take the form of componentwise multiplications, i.e.

$$\begin{aligned} H &\leftarrow H \circ F_H(X, W, H) \\ W &\leftarrow W \circ F_W(X, W, H) \end{aligned} \quad (1.28)$$

where $\circ$ is the Hadamard (component-wise) product between two matrices and where F_W and F_H are non-negative when X , W , and H are. When considering a Frobenius normed error, the Karush-Kuhn-Tucker first-order optimality conditions for $\mathbf{l}_2$ -NMF are given by

$$W \geq 0, \quad H \geq 0 \quad (1.29)$$

$$\nabla_W \|X - WH\|_F^2 \geq 0, \quad \nabla_H \|X - WH\|_F^2 \geq 0 \quad (1.30)$$

$$W \circ \nabla_W \|X - WH\|_F^2 = 0, \quad H \circ \nabla_H \|X - WH\|_F^2 = 0 \quad (1.31)$$

where

$$\nabla_W \|X - WH\|_F^2 = -2(X - WH)H^\top \quad (1.32)$$

$$\nabla_H \|X - WH\|_F^2 = -2W^\top(X - WH) \quad (1.33)$$

Combining (1.31) with (1.32) and (1.33), we get the following algorithm to approach problem $\mathbf{l}_2$ -NMF where, following the notations chosen in [GG08a], the Hadamard division is denoted by $[\cdot]/[\cdot]$.

Algorithm 1. Multiplicative Updates [MU]

Input : $X \in \mathbb{R}_+^{m \times n}$ factorization rank r and initialization type $\mathcal{I}$.

Output : $(W, H) \geq 0$: A rank- r $\mathbf{l}_2$ -NMF of $X \approx WH$.

```

1:  $[W_0, H_0] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
2: for  $k = 0, 1, 2, \dots$  do
3:    $H_{k+1} = H_k \circ \frac{[W_k^\top X]}{[W_k^\top W_k H_k]}$ 
4:    $W_{k+1} = W_k \circ \frac{[X H_k^\top]}{[W_k H_k H_k^\top]}$ 
5: end for
    
```

MU has been proven to be non-increasing [LS00]. However, convergence to a first-order stationary point is not guaranteed [Ber+07]. Fortunately, a slight modification pro-

1 | Preliminaries and background

posed in [Lin07] achieves this property. By denoting $[\cdot]_\epsilon = \max(\epsilon, \cdot)$ we obtain the following result.

Theorem 1.14 ([GG08a]). For every constant $\epsilon > 0$, the Frobenius norm $\|X - WH\|_F$ is non-increasing under the *modified multiplicative update*

$$H_{k+1} = \left[H_k \circ \frac{[W_k^\top X]}{[W_k^\top W_k H_k]} \right]_\epsilon \quad (1.34)$$

$$W_{k+1} = \left[W_k \circ \frac{[XH_k^\top]}{[W_k H_k H_k^\top]} \right]_\epsilon \quad (1.35)$$

for any $(W_k, H_k) \geq \epsilon$. Moreover, every limit point of this algorithm is a stationary point of the following optimization problem.

$$\min_{W \geq \epsilon, H \geq \epsilon} \|X - WH\|_F^2 \quad (1.36)$$

The **MU** method has the advantage to be easy to implement and to produce good results. However, there are some issues that need to be solved. On the one hand, since **MU** is a first-order rescaled gradient descent method [Gil14], i.e.

$$H_{k+1} = H_k - \frac{[H_k]}{[W_k^\top W_k H_k]} \circ \nabla_H \|X - W_k H_k\|_F^2 \quad (1.37)$$

$$W_{k+1} = W_k - \frac{[W_k]}{[W_k H_k H_k^\top]} \circ \nabla_W \|X - W_k H_k\|_F^2 \quad (1.38)$$

its convergence speed is slow. On the other hand, the method may have a zero denominator during iteration.

Note that, since the multiplicative updates are performed simultaneously on the two factors, it can be considered as both a full-space and block-coordinate search. A straightforward avenue for improvement is here to break the simultaneity of the updates and consider the intermediate improvements on one factor when updating the other [GG08a].

1.4.4 Alternating (non-negative) least squares [ANLS, ALS]

Coming back to the block-coordinate approach, the first intuitive way to go is to consider the largest possible blocks, here W and H independently, to implement the alternating optimization framework [PT94; SGB05]. We thereby obtain a *two-block coordinate descent* (2-BCD) method where the two convex sub-problems can be solved exactly at iteration k .

Algorithm 2. Alternating non-negative least squares [ANLS]

Input : $X \in \mathbb{R}_+^{m \times n}$ factorization rank r and initialization type $\mathcal{I}$.

Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $[W_0, H_0] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
2: for  $k = 0, 1, 2, \dots$  do
3:    $H_{k+1} = \arg \min_{H \in \mathbb{R}_+^{r \times n}} \|X - W_k H\|_F^2$  [P1]
4:    $W_{k+1} = \arg \min_{W \in \mathbb{R}_+^{m \times r}} \|X - W H_{k+1}\|_F^2$  [P2]
5: end for
    
```

Sub-problems [P1] and [P2], also referred to as *non-negative least square* (NNLS) problems, are equivalent to the two K_Means sub-problems (1.27) except from the ‘column-stochastic binary constraint’ on coefficients H . From that viewpoint, sub-problem [P1] amounts to finding the best combination of centroids to approximate each data point and in sub-problem [P2] the r centroids are moved to improve the overall approximation quality given the current weighting.

Some active-set methods have been designed specifically to address these sub-problems [KP08; KP11b]. Akin to the simplex method for linear optimization, active-set methods iteratively update the active set via pivoting (i.e. entering and removing variables from the active set) that guarantees the objective function to decrease [LH95]. Even if convergence to a stationary point can be guaranteed [GS00, Corollary 2], as r gets larger, ANLS tends to be rather slow compared to more sophisticated iterative methods that do not attempt to solve the sub-problems exactly.

A first attempt to obtain better performances was to relax the two sub-problems’ non-negativity constraints and instead apply them to the solution obtained from their unconstrained version, i.e. project the solutions on the positive orthant. This idea gave birth to the following *alternating least squares* (ALS) heuristic.

Algorithm 3. Alternating least squares [ALS]

Input : $X \in \mathbb{R}_+^{m \times n}$ factorization rank r and initialization type $\mathcal{I}$.

Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $[W_0, H_0] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
2: for  $k = 0, 1, 2, \dots$  do
3:    $H_{k+1} = \max \left( 0, \arg \min_{H \in \mathbb{R}^{r \times n}} \|X - W_k H\|_F^2 \right)$  [P1]
4:    $W_{k+1} = \max \left( 0, \arg \min_{W \in \mathbb{R}^{m \times r}} \|X - W H_{k+1}\|_F^2 \right)$  [P2]
5: end for
    
```

Sub-problems [P1] and [P2] can be very efficiently solved. Unfortunately, the ALS method comes with no theoretical guarantee and in fact often diverges in practice, especially for dense input matrices [Gil20]. Therefore, it is not suitable for directly solving $\mathbf{l}_2$ -NMF. In real applications, this method is usually used for preprocessing original data, i.e. initializing the data, especially for sparse matrices [CP09].

1.4.5 Hierarchical alternating least squares [HALS]

Cichoki et al. [CZA07] and independently several other authors [HVB11; GG08b] have proposed to even further decomposing the $\mathbf{l}_2$ -NMF variables in smaller consistent blocks: rank-1 matrices. As observed in equation (1.13), the $\mathbf{l}_2$ -NMF objective function can be expressed as a sum of r independent rank-1 contributions.

$$\|X - WH\|_F^2 = \left\| X - \sum_{j=1}^r W(:,j)H(j,:)^\top \right\|_F^2 \quad (1.39)$$

This scheme has a major advantage over other methods : the rank-1 sub-problems

$$W_{k+1}(:,l) = \arg \min_{W(:,l) \in \mathbb{R}_+^m} \|R_l - W(:,l)H_k(l,:)\|_F^2 \quad \forall l \quad (1.40)$$

$$H_{k+1}(l,:) = \arg \min_{H(l,:) \in \mathbb{R}_+^n} \|R_l - W_{k+1}(:,l)H(l,:)\|_F^2 \quad \forall l \quad (1.41)$$

where $R_l = X - \sum_{j \neq l} W(:,j)H(j,:)$ can be optimally solved in closed form. Considering $[\cdot]_+ = \max(0, \cdot)$, one can state the following seminal theorem.

Theorem 1.15 ([HVB11]). If $[R_l^\top W(:,l)]_+ \neq 0$, then

$$H^*(l,:) = \frac{[R_l^\top W(:,l)]_+}{\|W(:,l)\|_2^2} \quad (1.42)$$

is the unique global minimizer of

$$\arg \min_{H(l,:) \in \mathbb{R}_+^n} \|R_l - W(:,l)H(l,:)\|_F^2 \quad (1.43)$$

Theorem 1.15 has of course a dual form holding for (1.40). This yields the following updating rules

$$H_{k+1}(l,:) = \frac{[R_l^\top W_k(:,l)]_+}{\|W_k(:,l)\|_2^2} \quad W_{k+1}(:,l) = \frac{[R_l H_{k+1}(l,:)]_+}{\|H_{k+1}(l,:)\|_2^2} \quad (1.44)$$

with which each rank-one contribution can be modified in a cyclic manner.

Algorithm 4. Hierarchical alternating least squares [HALS]

Input : $X \in \mathbb{R}_+^{m \times n}$ factorization rank r and initialization type $\mathcal{I}$.

Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $[W_0, H_0] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
2: for  $k = 0, 1, 2, \dots$  do
3:   %  $H$  update
4:   for  $l = 1, 2, \dots, r$  do
5:      $H_{k+1}(l, :) = \left[ \frac{W_k(:, l)^\top X - \sum_{j \neq l} (W_k(:, l)^\top W_k(:, j)) H_k(j, :)}{\|W_k(:, l)\|_2^2} \right]_+$  [P1_1]
6:   end for
7:   %  $W$  update
8:   for  $l = 1, 2, \dots, r$  do
9:      $W_{k+1}(:, l) = \left[ \frac{X H_{k+1}(l, :)^\top - \sum_{j \neq l} W_k(:, j) (H_{k+1}(j, :)^T H_{k+1}(l, :)^T)}{\|H_{k+1}(l, :)\|_2^2} \right]_+$  [P2_1]
10:  end for
11: end for
    
```

The R_l notation was here avoided because, in practice, for computational efficiency purposes, R_l is never computed.

As for [MU](#), a potential issue with this method is that, in the course of the optimization process, one of the vectors $W_k(:, l)$ or $H_k(l, :)$ and the corresponding rank-one factor $W_k(:, l) H_k(l, :)^T$ may become equal to zero. This can lead to numerical instabilities. A possible way to overcome this problem and guarantee the convergence to a stationary point is to replace the zero lower bounds by a small positive constant $\epsilon \ll 1$ [[CZA07](#)].

Theorem 1.16 ([[GG08a](#)]). For every constant $\epsilon > 0$, every limit point of a BCD with modified closed-form update rules

$$H_{k+1}(l, :) = \left[\frac{R_l^\top W_k(:, l)}{\|W_k(:, l)\|_2^2} \right]_\epsilon \quad W_{k+1}(:, l) = \left[\frac{R_l H_{k+1}(l, :)}{\|H_{k+1}(l, :)\|_2^2} \right]_\epsilon \quad (1.45)$$

is a stationary point of the following optimization problem.

$$\min_{W \geq \epsilon, H \geq \epsilon} \|X - WH\|_F^2 \quad (1.46)$$

It is here also worth mentioning that in addition to convergence, unlike many other methods, Theorem 1.15 implies that [HALS](#) is monotonic.

HALS *outperforms* MU

The computational cost per iteration for HALS is the same as for MU, namely $\mathcal{O}(mnr)$ operations. However, there is evidence that HALS converges way faster. This can be explained by observing that the multiplicative updates (see (1.37) and (1.38)) are also modifying alternatively the columns of W and the rows of H but in a non-optimal way [GG08a].

$$\begin{aligned} H_{k+1}(l,:) &= H_k(l,:) \circ \frac{[W_k(:,l)^\top X]}{[W_k(:,l)^\top W_k H_k]} \\ W_{k+1}(:,l) &= W_k(:,l) \circ \frac{[X H_k(l,:)]^\top}{[W_k H_k H_k(l,:)]^\top} \end{aligned} \quad (1.47)$$

So, in addition to the slowness resulting from the simultaneous updates (see section 1.4.3), it becomes clear from 1.15 that MU is also sub-optimal whereas HALS involves an optimal update rule.

1.4.6 Faster HALS [A-HALS, E-A-HALS]

Given its growing popularity, researchers sought to improve even more the state-of-the-art HALS method.

Perform time-consuming steps sparingly

The first approach focused on taking advantage of the algebraic structure of the updates to enhance the ‘utility’ of each computational effort. In [GG12a], Gillis and Glineur propose to update each factor multiple times before updating the other such that better solutions of the sub-problems [P1_1] and [P2_1] can be obtained at a relatively cheap additional cost. This is possible since the factors $W_k(:,l)^\top W_k(:,j)$, $W_k(:,l)^\top X$ for [P1_1] and $H_{k+1}(j,:)H_{k+1}(l,:)^T$, $XH_{k+1}(l,:)^T$ for [P2_1] remain unchanged while sticking to the same sub-problem. The authors propose a heuristic criterion based on successive improvements to switch from one sub-problem to the other. The resulting method is denoted A-HALS.

Adding momentum

Since the HALS method can be regarded as a block projected gradient descent method [Gil20], i.e for [P1_1]

$$H_{k+1}(l,:) = \left[H_k(l,:) - \frac{1}{\|W_k(:,l)\|_2^2} W_k(:,l)^\top (W_k(:,l) H_k(l,:) - R_l) \right]_+ \quad (1.48)$$

it is very tempting to transfer to some extent the powerful extrapolation scheme used to accelerate gradient methods in convex optimization to HALS. In [AG19], a extrapolation scheme was applied successfully to address the l_2 -NMF problem.

Both accelerations can be combined on top of HALS resulting in the E-A-HALS method.

1.4.7 Initialization of the factors

Given the iterative nature of the presented local search numerical methods, they all require an initialization of the non-negative factors: W_0 and H_0 . As one might expect, the choice of initial values significantly affects the convergence speed of the algorithm and the quality of the solution to which it converges. Unfortunately, most initialization strategies come with no theoretical guarantee (e.g., a bound on the distance of the initial point to optimality) which can be explained in part by the complexity of the problem (see section 1.2.2) [Gil14].

In [Esp21], the author proposes to classify the initialization mechanisms for iterative NMF algorithms into three classes, referring to the main paradigm on which the initialization is based : (i) random-based schemes, (ii) structured initializations, (iii) evolutionary and nature-based mechanisms

Random-based initialization. The simplest way to choose the initial factors W_0 and H_0 is to construct them from random numbers. They require a low computational cost but can vary a lot in terms of quality. Moreover, two design parameters can have a big impact:

- The sparsity level can play a role since we expect good factorizations to be sparse (see section 1.1.2) and the boundary of the feasible domain only contains sparse matrices. In fact, generating dense initial matrices starts the exploration relatively far away from that boundary where solutions are in general located [Van+16].
- Randomness goes hand-in-hand with a chosen probability distribution. Typically, the simple uniform or a truncated Gaussian distribution is doing the job.

Structured initialization. Schemes belonging to the structured initialization class are based on the idea of applying near-separable NMF, low-rank factorizations or well-known clustering mechanisms on the data matrix X to generate initial factors for NMF.

- *Near-separable NMF* approximates the matrix X as $X(:, \mathcal{K})H$ where $\mathcal{K} \subseteq \{0, 1, \dots, n\}$ contains r indices and $H \geq 0$. Greedy algorithms such as SPA and SNPA can quickly (in polynomial time) produce an index set $\mathcal{K}$ such that $X(:, \mathcal{K})H \approx X$, even when X does not admit a separable factorization [Gil20].
- *Low-rank factorizations*: They include schemes using the SVD and more elaborate variants that originate from it. These strategies are totally deterministic in the sense that no randomization steps are involved.
- *Clustering mechanisms*: They include schemes using K_Means and more elaborate variants that originate from it. These strategies reduce the minimum use of random numbers.

Evolutionary and Natural Based Initialization. Schemes based on stochastic global search and evolutionary optimization methods can also be used as feeding schemes for NMF algorithms. Under this class, we can enumerate a number of different population-based algorithms: Genetic Algorithms, Particle Swarm Optimization, Fish School Search, Differential Evolution and Fireworks Algorithm [JT11b; JT11a; Sta+07].

Note that different local search methods for NMF can be combined to result in methods with an ever greater potential (see heuristics [HYB](#) and [C_HYB](#)). In such cases, one could argue that these heuristics are used as initialization schemes for one another.

2

Existing algorithmic approaches for ExactNMF

SINCE the early days of NMF, the linear algebra community has continually sought to improve the optimization methods to deal with the challenging NMF problem in all its forms. This chapter aims at introducing the state-of-the-art algorithmic approaches that have been proposed to tackle the [ExactNMF](#) problem.

Speed VS Quality: Two distinct objectives

While designing optimization strategies to address a given non-convex optimization problem as NMF, one can seek to achieve simultaneous different objectives. First of all, it goes without saying that since we minimize an error function, methods are expected to provide a **high-quality solution**, i.e. with the smallest possible error. In particular, in the [ExactNMF](#) context, we look for global minima, i.e. factorizations s.t. $\|X - WH\| = 0$. On the contrary, in a majority of settings where the approximate NMF applies, 'only' *acceptable* solutions are expected rather than the *optimal* one.

Speaking of the optimality of the generated solutions, it is very important to keep in mind that **convergence guarantees** are also a critical pledge of quality, especially in a local search framework.

On the other hand, we always have to take the available computational resources into consideration. Coming up with computationally untractable methods can at most capitalize on further computing power improvements (see section 1.3). It is therefore of utmost importance to develop the most **effective/cheap methods** possible.

It becomes clear that a clever trade-off between speed and quality must be found in the [ExactNMF](#) context.

2.1 BCD heuristics for `ExactNMF`

As mentioned in section 1.2.1, even though some promising algorithmic theoretical guarantees have been obtained, they do not directly translate into practical algorithms, even when dealing with very small matrices. The linear algebra community has so far failed to propose algorithms specifically designed to directly tackle the `ExactNMF` problem. Most numerical algorithms have been developed for approximation problems such as ℓ_2 -NMF (see section 1.4).

It is still possible to build on previously presented algorithmic approaches to compute exactNMFs. In fact, if such an algorithm obtains a reconstruction error equal to zero, i.e. $X = WH$, then it has solved the corresponding `Fixed_ExactNMF` problem.

Since in the `ExactNMF` context, the balance is tipped more towards quality than speed, more time-consuming **globalization heuristics** have been proposed to explore the solution space more thoroughly and thereby augment the chance to find exact factorizations.¹

Remark 2.1 (Are the factorizations found really exact?). *As it was rightly emphasized in [Van+16], all the numerical experiments performed in this thesis are with floating point arithmetic. Hence, a given factorization will be considered as exact if the relative error $\|X - WH\|_F / \|X\|_F \leq \gamma$. So, the computed factorizations are not exact but high-precision solutions. We here arbitrarily fix the threshold $\gamma = 10^{-6}$ which seems enough to suppose that the factorization found will directly lead to an exact factorization. In this setting, factorization can at best be computed up to machine precision. It would therefore also be useful to develop some rounding strategies to transform a high-accuracy solution into an exactNMF, when possible [Van+16]. Also note that from now on, the term 'error' will directly refer to a relative error which makes more sense when considering matrices with arbitrary scaling.*

2.1.1 Multi-start heuristics [MS,MBS]

A first intuitive approach arises from the observation that, since the ℓ_2 -NMF problem can be highly non-convex, the quality of the output factorization is very closely related to the quality of the initialization.

A natural way to proceed would then be to apply the full local search optimization process K times starting from different initial points and keep the factorization achieving the smallest error (see Algorithm 5). This *multi-start* (MS) heuristic was for example proposed in [GG12b] under the name "MS1".

¹Note that, based on the discussion in [Van+16], all the heuristics presented in this section will be initialized by sparse random factors.

Algorithm 5. Multi-start heuristic [MS]**Input :** $X \in \mathbb{R}_+^{m \times n}$, $0 < r \leq \min(m, n)$, K and init type $\mathcal{I}$.**Output :** $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $e \leftarrow 1$ 
2: for  $i = 1, \dots, K$  do
3:    $[\tilde{W}_0, \tilde{H}_0] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
4:    $[\tilde{W}, \tilde{H}] \leftarrow \text{BCD\_refinement}(X, \tilde{W}_0, \tilde{H}_0)$ 
5:    $\tilde{e} \leftarrow \|X - \tilde{W}\tilde{H}\|_F / \|X\|_F$ 
6:   if  $\tilde{e} < e$  then
7:      $(W, H) \leftarrow (\tilde{W}, \tilde{H}), e \leftarrow \tilde{e}$ 
8:   end if
9: end for

```

The `BCD_refinement` function implements one of the state-of-the-art BCD algorithms (e.g. A-HALS) which is subjected to a threshold but also an improvement termination criteria to stop the optimization process when convergence is observed. Note that for all the heuristics presented in this thesis, the maximum number of refinement iterations is fixed at 2000000.

One could here argue that a lot of computational effort is carried out in vain: we should be able to assess whether an optimization process is promising before convergence. Unlike the **MS**, the *multi-best-start* (MBS) heuristic performs K starts but exploits each of them by N local search iterations. Only the best factorization is then finally refined (see Algorithm 6).

Algorithm 6. Multi-best-start heuristic [MBS]**Input :** $X \in \mathbb{R}_+^{m \times n}$, $0 < r \leq \min(m, n)$, K , N and init type $\mathcal{I}$.**Output :** $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $e \leftarrow 1$ 
2: for  $i = 1, \dots, K$  do
3:    $[\tilde{W}_0, \tilde{H}_0] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
4:    $[\tilde{W}, \tilde{H}] \leftarrow \text{BCD\_search}(X, \tilde{W}_0, \tilde{H}_0, N)$ 
5:    $\tilde{e} \leftarrow \|X - \tilde{W}\tilde{H}\|_F / \|X\|_F$ 
6:   if  $\tilde{e} < e$  then
7:      $(W, H) \leftarrow (\tilde{W}, \tilde{H}), e \leftarrow \tilde{e}$ 
8:   end if
9: end for
10:  $[W, H] \leftarrow \text{BCD\_refinement}(X, W, H)$ 

```

The `BCD_search` function implements one of the state-of-the-art BCD algorithms (e.g. A-HALS). We refer to [Van+16] for the choice of the hyper-parameters K and N . Unfortunately, for some difficult matrices, multi-start strategies are unable to find exact factorizations. Even more elaborated search strategies are needed.

2.1.2 Rank-by-rank heuristic [RBR]

In [Van+16], the authors propose a heuristic with the aim to construct progressively an exactNMF. Starting from an optimal rank-1 solution (see section 1.2.2), the other dimensions are added to the optimization process one at a time. For each of them, K attempts with N iterations of a state-of-the-art BCD algorithm (e.g. A-HALS) are performed to find the optimal approximation of a given rank before augmenting the factorization rank (see Algorithm 7).

Algorithm 7. Rank-by-rank heuristic [RBR]

Input : $X \in \mathbb{R}_+^{m \times n}$, r , K , N .
Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $[u_1, \sigma_1, v_1] \leftarrow \text{SVD}(X)$ 
2:  $(W_1, H_1) \leftarrow (\sqrt{\sigma_1}|u_1|, \sqrt{\sigma_1}|v_1^\top|)$  % See section 1.2.2
3: for  $k = 2, \dots, r$  do % Add one dimension at the time.
4:    $e_{\min} \leftarrow 1$ 
5:   for  $i = 1, \dots, K$  do
6:      $(\tilde{W}_k(:, 1:k-1), \tilde{H}_k(1:k-1, :)) \leftarrow (W_{k-1}, H_{k-1})$ 
7:      $(\tilde{W}_k(:, k), \tilde{H}_k(k, :)) \leftarrow \text{Rand\_Initialization}(m, n, 1)$ 
8:      $(\tilde{W}_k, \tilde{H}_k) \leftarrow \text{BCD\_search}(X, \tilde{W}_k, \tilde{H}_k, N)$ 
9:      $\tilde{e} \leftarrow \|X - \tilde{W}_k \tilde{H}_k\|_F / \|X\|_F$ 
10:    if  $\tilde{e} < e_{\min}$  then
11:       $(W_k, H_k) \leftarrow (\tilde{W}_k, \tilde{H}_k)$ ,  $e_{\min} \leftarrow \tilde{e}$ 
12:    end if
13:  end for
14: end for
15:  $[W, H] \leftarrow \text{BCD\_refinement}(X, W_r, H_r)$ 

```

2.1.3 Simulated annealing heuristic [SA]

Another proposition made by Vandaele et al. [Van+16] follows the widely used simulated annealing framework [Pir96]. The idea, implemented in Algorithm 8, is to explore the neighbourhood of the initial solution in a random fashion with the hope to find a better solution. This local exploration is performed with P decreasing acceptance levels (given by temperatures between T_0 and T_{end}) with K attempts for each level.

1. **Random drift** : Select a small subset $\mathcal{J}$ of J rank-one factors $W(:, \mathcal{J})H(\mathcal{J}, :)$ at random and reinitialize these rank-one factors randomly.
2. **Local search** : Improve the corresponding solution locally with N iterations of a state-of-the-art BCD algorithm (e.g. A-HALS).
3. **Acceptance of the proposal**: Decide whether to keep the refined neighbouring solution depending on its error and on the current temperature. Note that solution whose error is smaller than the error of the current solution is always kept.

Algorithm 8. Simulated annealing heuristic [SA]

Input : $X \in \mathbb{R}_+^{m \times n}$, $r, K, N, J, \gamma, T_0, T_{end}, P$ and init type $\mathcal{I}$.

Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $[W_\#, H_\#] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
2:  $e \leftarrow \|X - W_\# H_\#\|_F / \|X\|_F$ ,  $e_{min} \leftarrow e$ 
3:  $T \leftarrow T_0$ 
4: while  $T > T_{end}$  do
5:   for  $i = 1, \dots, K$  do
6:      $(\tilde{W}_\#, \tilde{H}_\#) \leftarrow (W_\#, H_\#)$ 
7:      $\mathcal{J} \leftarrow \text{pick randomly } J \text{ indices in } \{1, \dots, r\}$ 
8:      $[\tilde{W}_\#(:, \mathcal{J}), \tilde{H}_\#(\mathcal{J}, :)] \leftarrow \text{Rand\_Initialization}(m, n, J)$ 
9:      $[\tilde{W}_\#, \tilde{H}_\#] \leftarrow \text{BCD\_Search}(X, \tilde{W}_\#, \tilde{H}_\#, N)$ 
10:     $\tilde{e} \leftarrow \|X - \tilde{W}_\# \tilde{H}_\#\|_F / \|X\|_F$ 
11:    if  $\tilde{e} < e$  or  $U[0, 1] < \exp(-\frac{e - \tilde{e}}{T})$  then
12:       $(W_\#, H_\#) \leftarrow (\tilde{W}_\#, \tilde{H}_\#)$ ,  $e \leftarrow \tilde{e}$ 
13:      if  $e < e_{min}$  then
14:         $(W, H) \leftarrow (W_\#, H_\#)$ ,  $e_{min} \leftarrow e$ 
15:        if  $e_{min} < \gamma$  then
16:           $T = T_{end}$ , break
17:        end if
18:      end if
19:    end if
20:  end for
21:   $T \leftarrow T - (T_0 - T_{end})/P$ 
22: end while
23:  $[W, H] \leftarrow \text{BCD\_refinement}(X, W, H)$ 

```

The best-identified solution is finally refined to obtain an exact factorization (see Algorithm 8). It is important to note that the SA heuristic highly encourages exploration since it allows for solutions with higher errors to be explored (although the probability for this to happen goes to zero as the temperature decreases). In that sense, SA can, to some extent, be seen as a (partial) multi-start heuristic.

Note that in [Van+16], the authors propose a hybridization [HYB] where the SA heuristic is initialized by the RBR approach.

2.2 Conic-based heuristics for **ExactNMF**

Of course, one could try to elaborate even more efficient methods from the same state-of-the-art BCD algorithms to address the **ExactNMF** problem. Even if the heuristics presented in [Van+16] are sufficient in a wide variety of cases, there is still a place for improvement in particular for especially difficult settings. In [Lep+21], Leplat et al. got off the beaten track and introduce two *conic optimization* reformulations to address the $\mathbf{l}_2$ -NMF problem and by extension the **ExactNMF**. In fact, in doing so, they opened the way to new algorithmic approaches to computing high-quality local minima for the NMF optimization problem without relying on the BCD framework. Already in this founding paper, they appeared to be a promising complement to the heuristics presented in section 2.1.

2.2.1 NMO and NMU with conic optimization

The key idea in [Lep+21] is to take advantage of a change in the structure of the NMF problem while keeping the main goal intact, i.e. finding an exact factorisation $X = WH$. More concretely, the minimization of the least square approximation error in $\mathbf{l}_2$ -NMF can be reformulated on the one side by the maximization of the $\mathbf{l}_1$ -norm of a pair of underapproximation factors, i.e.

Problem 2.2 (NMU). Given a non-negative matrix $X \in \mathbb{R}_+^{m \times n}$ and a factorization rank r , compute *non-negative matrix underapproximation* (NMU) involving two non-negative matrices $W \in \mathbb{R}_+^{m \times r}$ and $H \in \mathbb{R}_+^{r \times n}$ such that

$$\begin{aligned}
& \max_{W \in \mathbb{R}^{m \times r}, H \in \mathbb{R}^{r \times n}} \sum_{f \in \mathcal{M}} \sum_{g \in \mathcal{G}} \left(\sum_{k \in \mathcal{R}} W(f, k) H(k, g) \right) \\
& \text{s.t.} \quad \sum_{k \in \mathcal{R}} W(f, k) H(k, g) \leq X(f, g) \quad \forall f \in \mathcal{M}, g \in \mathcal{G} \\
& \quad \quad W(f, k) \geq 0, H(k, g) \geq 0 \quad \forall f \in \mathcal{M}, g \in \mathcal{G}, k \in \mathcal{R}
\end{aligned} \tag{2.1}$$

where $\mathcal{M} = \{1, \dots, m\}$, $\mathcal{G} = \{1, \dots, n\}$ and $\mathcal{R} = \{1, \dots, r\}$. On the other side, $\mathbf{l}_2$ -NMF can be reformulated by the minimization of the $\mathbf{l}_1$ norm of a pair of overapproximation factors, i.e.

Problem 2.3 (NMO). Given a non-negative matrix $X \in \mathbb{R}_+^{m \times n}$ and a factorization rank r , compute *non-negative matrix overapproximation* (NMO) involving two non-negative matrices $W \in \mathbb{R}_+^{m \times r}$ and $H \in \mathbb{R}_+^{r \times n}$ such that

$$\begin{aligned}
 & \min_{W \in \mathbb{R}^{m \times r}, H \in \mathbb{R}^{r \times n}} \sum_{f \in \mathcal{M}} \sum_{g \in \mathcal{G}} \left(\sum_{k \in \mathcal{R}} W(f, k) H(k, g) \right) \\
 & \text{s.t.} \quad \sum_{k \in \mathcal{R}} W(f, k) H(k, g) \geq X(f, g) \quad \forall f \in \mathcal{M}, g \in \mathcal{G} \\
 & \quad W(f, k) \geq 0, H(k, g) \geq 0 \quad \forall f \in \mathcal{M}, g \in \mathcal{G}, k \in \mathcal{R}
 \end{aligned} \tag{2.2}$$

Conic formulation for **NMU**

The **NMU** problem is nonconvex because of the bilinear terms appearing in the objective and the constraint $WH \leq V$. By an appropriate change of variables $W(f, k) = e^{U(f, k)}$ and $H(k, g) = e^{V(k, g)}$ which incorporates the non-negativity constraints, we can rewrite (2.1) as a maximization problem of a convex function (logarithm of the sums of exponentials) over a convex set²,

$$\begin{aligned}
 & \max_{U \in \mathbb{R}^{m \times r}, V \in \mathbb{R}^{r \times n}, t \in \mathbb{R}^{m \times r \times n}} \log \left(\sum_{f, g, k} e^{U(f, k) + V(k, g)} \right) \\
 & \text{s.t.} \quad \sum_{k \in \mathcal{R}} t(f, k, g) \leq X(f, g) \quad \forall f \in \mathcal{M}, g \in \mathcal{G} \\
 & \quad [t(f, k, g), 1, U(f, k) + V(k, g)] \in K_{\text{exp}} \quad \forall f \in \mathcal{M}, g \in \mathcal{G}, k \in \mathcal{R}
 \end{aligned} \tag{2.3}$$

where $K_{\text{exp}} \subset \mathbb{R}^3$ denotes the (primal) exponential cone defined as

$$K_{\text{exp}} = \left\{ (x_1, x_2, x_3) \mid x_1 \geq x_2 e^{\frac{x_3}{x_2}}, x_2 > 0 \right\} \cup \left\{ (x_1, 0, x_3) \mid x_1 \geq 0, x_3 \leq 0 \right\} \tag{2.4}$$

Conic formulation for **NMO**

The **NMO** problem is also nonconvex because of the bilinear terms appearing in the objective and the constraint $WH \geq V$. By an appropriate change of variables $W(f, k) = \sqrt{U(f, k)}$ and $H(k, g) = \sqrt{V(k, g)}$ which incorporates the non-negativity constraints, we can rewrite (2.1) as a minimization problem of a concave function over a convex set²,

²For complete developments, please refer to [Lep+21]. Note that the maximization (resp. minimization) of the given convex (resp. concave) function on a convex domain remains a non-convex problem!

$$\begin{aligned}
& \min_{W \in \mathbb{R}^{m \times r}, H \in \mathbb{R}^{r \times n}, t \in \mathbb{R}^{m \times r \times n}} \sum_{f \in \mathcal{M}} \sum_{g \in \mathcal{G}} \left(\sum_{k \in \mathcal{R}} \sqrt{U(f, k)} \sqrt{V(k, g)} \right) \\
& \text{s.t.} \quad \sum_{k \in \mathcal{R}} t(f, k, g) \geq X(f, g) \quad \forall f \in \mathcal{M}, g \in \mathcal{G} \\
& \quad [U(f, k), \frac{1}{2}T(k, g), t(f, k, g)] \in Q_r^3 \quad \forall f \in \mathcal{M}, g \in \mathcal{G}, k \in \mathcal{R}
\end{aligned} \tag{2.5}$$

where Q_r^3 denotes the 3-dimensional rotated second-order cone defined as

$$Q_r^3 = \left\{ (x_1, x_2, x_3) \mid 2x_1x_2 \geq x_3^2, x_1 \geq 0, x_2 \geq 0 \right\} \tag{2.6}$$

2.2.2 IPM-based Frank-Wolfe algorithm for **NMO/NMU**

To tackle both the **NMU** and **NMO** problems, starting from the fact that both problems can be written as the minimization of a concave function Φ over a convex set denoted by Q , the authors of [Lep+21] propose an iterative algorithm where, at each stage, the original minimization problem is replaced by a (presumably) simpler problem in which the local linear approximation to Φ , obtained from its first order Taylor expansion at the current iterate, is minimized over Q . Concretely, the i th iterate denoted $Z^{(i)} = (U^{(i)}, T^{(i)})$ is obtained in the following way

$$\begin{aligned}
Z^{(i)} & \in \arg \min_{Z \in Q} \Phi(Z^{(i-1)}) + \left\langle \nabla \Phi(Z^{(i-1)}), Z - Z^{(i-1)} \right\rangle \\
& \in \arg \min_{Z \in Q} \left\langle \nabla \Phi(Z^{(i-1)}), Z \right\rangle
\end{aligned} \tag{2.7}$$

This is a particular case (with $\tau^{(i)} = 1$) of the more general *Frank-Wolfe* (FW) algorithm [FW56],

Algorithm 9. Frank-Wolfe algorithm [FW]

Input : $Z^{(0)} \in Q$.

- 1: **for** $i = 1, \dots, \text{maxiter}$ **do**
- 2: Compute $S^{(i)} = \arg \min_{Z \in Q} \left\langle \nabla \Phi(Z^{(i-1)}), Z \right\rangle$ [LMO]
- 3: Choose $0 < \tau^{(i)} \leq 1$
- 4: Update $Z^{(i)} = (1 - \tau^{(i)}) Z^{(i-1)} + \tau^{(i)} S^{(i)}$
- 5: **end for**

which is a well-known first-order method that relies on the ability to compute efficiently the so-called *Linear Minimization Oracle* [LMO]. Since problems **NMU** and **NMO** were

reformulated as convex conic optimization problems, the sub-problems **LMO** can be successfully solved to optimality with an interior-point method (IPM), as available for example in MOSEK [ApS19]. The conic-based version of the **FW** algorithm with $\tau^{(i)} = 1$ will here be denoted [CFW].

Convergence

In terms of convergence, considering the so-called *FW gap* of Φ at $Z^{(i)}$ which is a standard measure of stationarity in the analysis of FW algorithms [Lac16],

$$\mu^{(i)} = \min_{Z \in Q} \left\langle \nabla \Phi(Z^{(i-1)}), Z - Z^{(i)} \right\rangle \quad (2.8)$$

the **FW** algorithm with $\tau^{(i)} = 1$ applied to **NMU** and **NMO** converges to an approximate stationary point with gap smaller than ϵ in at most $\mathcal{O}(1/\epsilon)$ iterations.

Theorem 2.4 ([Lep+21]). Consider the problem

$$\min_{Z \in Q} \Phi(Z) \quad (2.9)$$

where Φ is a continuously differentiable concave function over the compact convex domain Q . Let us denote $Z^{(i)}$ the sequence of iterates generated by the **FW** algorithm applied on the latter minimization problem. Assume there exists a constant $\tilde{\tau} > 0$ such that $\tilde{\tau} \leq \tau^{(i)} \leq 1$ for all i . Then the minimal FW gap, defined as $\tilde{\mu}^{(i)} = \min_{0 \leq j \leq i} \mu^{(j)}$ satisfies for all $i = 1, 2, \dots$

$$\tilde{\mu}^{(i)} \leq \frac{1}{\tilde{\tau}} \frac{\Phi(Z^{(0)}) - \Phi^*}{i + 1} \quad (2.10)$$

where $\Phi(Z^{(0)}) - \Phi^*$ is the (global) accuracy of the initial iterate.

Note that the compactness of Q in problems **NMU** and **NMO** can be ensured under mild modifications, see discussion in [Lep+21].

Sparsity pattern integration (SPI)

Even if convergence is ensured, another annoying problem arises with the new conic-based approach. In some settings, the required level of accuracy cannot be reached even if the solution is close to convergence. This is caused by a numerical limitation of the **FW** solver.

Considering the **NMO** case³, the situation can become problematic when some variable tends to zero. In fact, the corresponding entry of the gradient of ϕ tends to ∞ which therefore ends the optimization process.

In order to tackle this issue and enable the solution to reach the desired tolerance, Lepat et al. [Lep+21] decided to add a *Sparsity Pattern Integration* (SPI) stage close to the

³A similar reasoning can also be applied to the **NMU** case.

end of the **FW** optimization process. The SPI procedure puts to 0 all the variables below a certain threshold δ_{SPI} . These variables are then removed from the optimization problem and are therefore (definitively) integrated into the sparsity pattern. The effectiveness of this strategy is illustrated in Figure 2.1. Note that the method composed of the **CFW** algorithm and the final SPI refinement will be denoted **C_SPI**.

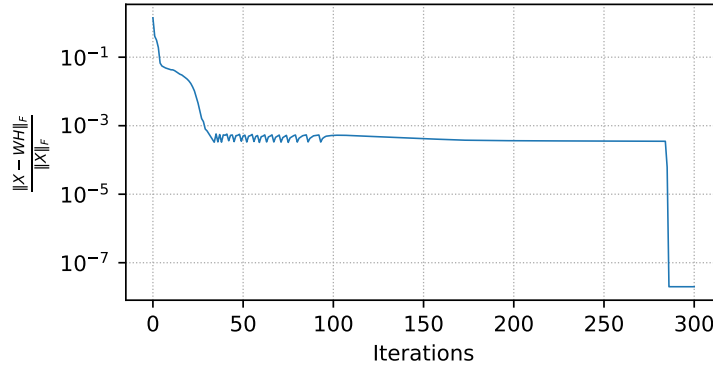


Figure 2.1 Illustration of the beneficial effect of the SPI procedure for matrix M_2 (see section 5.2.1) and $r = 3$ with $\delta_{SPI} = 10^{-6}$ on the relative approximation error.

Full-space search

One of the main characteristics of the **FW** algorithm is that all the variables of the problem are optimized simultaneously. Since we optimize them on a linearization at each iteration, in comparison with the other full-space **MU** method described in section 1.4, it is clear that the **FW** algorithm enjoys a greater mobility within the search space. In fact, at each iteration, the solution will follow the descent direction provided by the linearization of the objective function until a border of the solution space is reached. There is no structural limitation in the method to moderate the progress in the direction of the search. This interesting exploration behavior is illustrated in Figure 2.2.

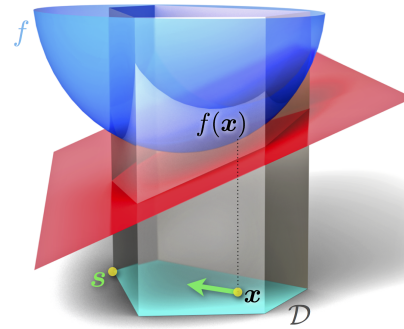


Figure 2.2 Illustration of the mobility capabilities of the **FW** algorithm [Wik22a]. Note that in the NMF setting of this thesis, f is concave.

Scalability

It is here important to mention that since the **FW** algorithm relies on poorly scalable IPMs to tackle problems **NMU** and **NMO**, it has been observed that the conic approach is significantly slower (at this point) than state-of-the-art BCD algorithms to compute approximate NMFs. Barring major changes, the existing and new approaches involving the conic reformulations are of interest only in the **ExactNMF** context. In fact, they implement an interesting alternative for exploring the solution space in search of exact NMFs. Promising results were already exposed in [Lep+21].

Get rid of IPMs ?

Given the important scalability issue caused by the computational heaviness of IPMs, one could try to get rid of them by proposing an alternative way to exploit the **NMU** and **NMO** formulation to address the **ExactNMF** problem.

To do so, it is very tempting to try to transpose in some way the successful schemes (e.g. BCD) implemented for the NMF formulation to the **NMU** and **NMO** formulations. In that perspective, one idea could be to solve these problems iteratively by considering at each step an analytical solution to a given sub-problem involving a subset of the variables.

Since the under- or over-approximation constraints must be satisfied along the whole optimization process, starting from a ‘legal’ initialization, it will appear as an *analytic iterative deflation process* of both factors. Considering the l_1 -norm objectives of **NMU** and **NMO**, if full optimality is achieved at each iteration, the selected variables will be pushed to zero or activate one of the under- or over-approximation constraints and thereby impose quite brutally the sparsity pattern of the solution which then highly depends on the order of consideration of the variables.

A solution could be to consider at each iteration, only the most promising variable and modify it proportionally to a certain metric on the approximation error. This strategy unfortunately gives birth to another problem (opposite to the first). Since variables are considered on a rotating basis as a result of the selection, they are all limply deflated towards a very poor factorization with no sparsity.

This intuition was indeed verified in practice, see an example in Figure 2.3. As a result, it is here unclear how a trade-off between the two approaches could result in an effective method. We here conclude that considering sub-problems of **NMU** or **NMO** involving only subsets of the variables cannot apparently lead to an interesting alternative to avoid using IPMs. The effectiveness of the conic approach seems to rely on its full-space nature, i.e. all variables are needed at each iteration. This will become even clearer in section 4.3.3.

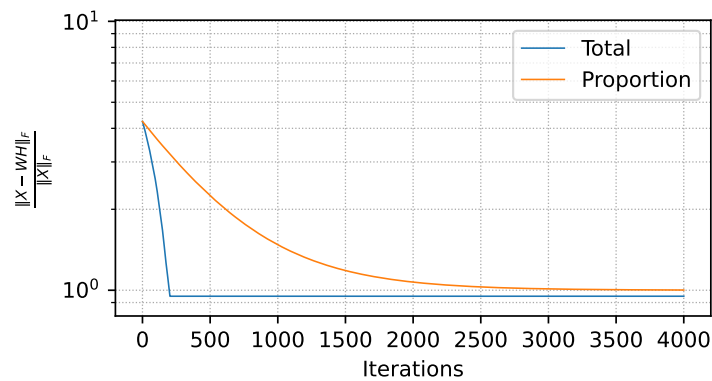


Figure 2.3 Illustration on S_{28} (see section 3.2.2) with $r = 10$ of the low potential of the *analytic iterative deflation* approach. In the *Total* approach, the considered elements are fully optimized. In the *Proportion* case, elements are proportionally modified.

Applications of ExactNMF

THIS section aims at illustrating some applications where ExactNMF arises and how it can help to solve some practical or theoretical problems. In particular, we will highlight some specific settings in which we will test the new conic-based heuristics introduced in this thesis.

3.1 Motivating applications for ExactNMF

In the machine learning and data mining context, it typically does not make sense to look for exactNMFs since the data is usually contaminated with noise. However, there exist multiple other applications where solving ExactNMF or finding a non-negative rank is particularly relevant. In this section, we provide a quick overview of these applications and a more in-depth insight into one of them which became predominant in the study of ExactNMF.

3.1.1 A non-exhaustive overview

Let us describe briefly four distinct contexts in which the ExactNMF comes directly or indirectly into play.

Joint probability distribution [SS93; CR10; Dew22].

Let A and B be two correlated random variables that can respectively take m and n distinct values and $P \in \mathbb{R}^{m \times n}$ the matrix representing their joint probability distribution. Let C be a discrete random variable explaining the correlation between A and B , i.e. A and B are independent conditional on C . By denoting $A_{(k)} \in \mathbb{R}^n$ and $B_{(k)} \in \mathbb{R}^m$ the conditional probability vector of A and B when $C = k$, their joint probability distribution matrix can be expressed as

$$P = \sum_{k=1}^{r_+} \mathbb{P}[C = k] A_{(k)} B_{(k)}^\top \quad (3.1)$$

where r_+ is the smallest support of the hidden random variable C which explains the correlation between the pair of discrete random variables and the non-negative rank of joint probability distribution matrix P .

Minimum biclique cover number of a bipartite graph [Van+16; Fio+13].

Let $G = (V = V_1 \cup V_2, E)$ be a bipartite graph. A complete bipartite subgraph of G , referred to as a biclique, is a subgraph $G' = (V'_1 \cup V'_2, E')$ such that all vertices in V'_1 and V'_2 are connected. The *minimum biclique cover number* $\text{bc}(G)$ of G is the minimum number of bicliques needed to cover all edges in G . By denoting the biadjacency matrix of g as $X_G \in \{0, 1\}^{m \times n}$, we have that

$$\text{bc}(G) \leq \text{rank}_+(X_G) \quad (3.2)$$

and so computing an exactNMF of X_G provides an upper bound for the minimum biclique cover of G .

Communication complexity [lovasz1989communicatio; Yao79; Yan88].

Let us consider a traditional communication setting where two parties, traditionally called Alice and Bob, must evaluate a known function $f : A \times B \rightarrow C$ for some inputs a (known by Alice) and b (known by Bob). To achieve their goal, they must exchange information by communicating a bit at a time following some protocol decided in advance. In that context, the *communication complexity* of f , denoted $D(f)$, is defined as the minimum number of bits of information that Alice and Bob need to exchange for the worst-case input. A communication matrix $M \in \mathbb{R}^{|A| \times |B|}$ such that $M_{ij} = f(i, j)$ can be associated with the problem. It can be shown that

$$\log(\text{rank}_+(M)) \leq D(f) \leq \log(\text{rank}(M)) \log(\text{rank}_+(M)) \quad (3.3)$$

Conjecturing new theoretical results on rank_+ or disproving them [Van+16; GG12b].

As any exactNMF of a non-negative matrix provides an upper bound on its non-negative rank, one can use this technique to find counter-examples to conjectures dealing with the non-negative rank of matrices or strengthen our belief that some conjectures are correct.

3.1.2 Extension complexity of polyhedra

In linear or more generally in combinatorial optimization, the convex hull of the set of feasible solutions is expressed by linear equalities and inequalities. All together, these constraints form a polyhedron. When considering a bounded polyhedron¹, namely a *polytope*

$$\mathcal{P} = \left\{ x \in \mathbb{R}^d \mid a_i^\top x \leq b_i, i = 1, \dots, N \right\} \quad (3.4)$$

¹Most results can be extended to unbounded polyhedra; see the discussion in [Fio+13].

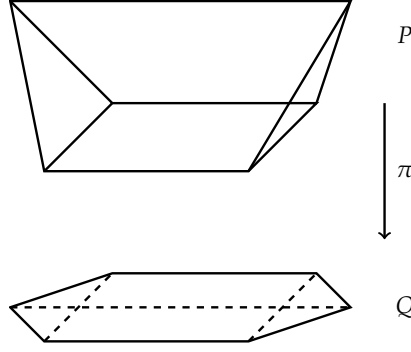


Figure 3.1 Minimum-size extended formulation of the regular hexagon with five facets. Figure taken from [Dew22].

a major issue consists in representing these polytopes compactly, i.e. with as few inequalities/facets as possible (possibly using additional variables). See Figure 3.1 for an archetypical example. To approach this problem more formally, we introduce the following definition.

Definition 3.1 ([Fio+13]). An *extension* or *extended formulation* of a polytope $\mathcal{P} \subseteq \mathbb{R}^d$ is a system of linear inequalities defining a polyhedron $\mathcal{Q} \subseteq \mathbb{R}^e$ along with an linear map $\pi : \mathbb{R}^e \mapsto \mathbb{R}^d$ satisfying $\pi(\mathcal{Q}) = \mathcal{P}$. The size of the extension is the number of inequalities in the system.

As exposed in [Van+16], if the number of facets of $\mathcal{P}$ is large (possibly growing exponentially with some dimension of the problem), a crucial question is whether there exists an extension with a small number of facets (ideally bounded by a polynomial), in which case a linear program over $\mathcal{P}$ can be solved much more effectively using an equivalent formulation over $\mathcal{Q}$. It is thereby relevant to introduce the following concept.

Definition 3.2. The minimum number of facets appearing in any extension of $\mathcal{P}$ is called the *extension complexity* of $\mathcal{P}$, denoted $\text{xc}(\mathcal{P})$.

In [Yan88], Yannakakis proved that there exists a direct link between the extension complexity of a polytope $\mathcal{P}$ and the non-negative rank of its slack matrix $\mathcal{S}_{\mathcal{P}} \in \mathbb{R}_+^{m \times n}$ defined by

$$\mathcal{S}_{\mathcal{P}}(i, j) = b_i - a_i^\top v_j \geq 0 \quad \forall i, j \quad (3.5)$$

where $v_1, \dots, v_n$ are the vertices of $\mathcal{P}$.

Theorem 3.3 ([Yan88]). Let $\mathcal{P}$ be a polytope of dimension larger than one, and let $\mathcal{S}_{\mathcal{P}}$ be its slack matrix. Then

$$\text{rank}_+(\mathcal{S}_{\mathcal{P}}) = \text{xc}(\mathcal{P}) \quad (3.6)$$

The latter theorem thus directly links the study of extension complexity with the study of non-negative rank and by extension the study of `ExactNMF`. Taking advantage of this new connection, many results concerning the extended formulations of various families of low-dimensional polytopes as well as specific polytopes associated with important optimization problems were obtained [KSZ22; Rot17].

Note that the slack matrix of a polytope is not unique since the inequalities can be scaled, and the rows and columns permuted, but this does not influence its non-negative rank.

3.2 (Open) Problems of interest

In this section, we present the difficult exactNMF settings on which we will test the new conic-based heuristics introduced in this thesis. They involve open problems and conjectures to reinforce or disprove.

3.2.1 Infinitesimally rigid non-negative factorizations

In [KK21], the uniqueness of non-negative matrix factorizations is studied using ideas from rigidity theory² in the case when the non-negative rank of a matrix is equal to its rank. In that context, *global rigidity* is defined as the uniqueness of an NMF, *local rigidity* as a necessary condition for global rigidity, and *infinitesimal rigidity* as a sufficient condition for local rigidity. The authors provide a characterization of infinitesimal rigidity that can be checked computationally [KK21, Proposition 3.3]. This result leads to an algorithm for determining the local rigidity of an NMF which introduces the strongest necessary condition for the uniqueness of an NMF so far.

In their paper, the authors generate 15 infinitesimal non-negative factorizations and conjecture that they are in fact globally rigid. Note that it is also mentioned that, for 2 of them, 10 runs of the `SA` heuristic were unable to recover the supposedly unique factorization.

3.2.2 Extension complexity of n -gons

The study of extended formulations and extension complexity has a rich history (see [CCZ10]) and has enjoyed particular attention over the last decade. Building on Yannakakis' seminal theorem 3.3 and on always more advanced geometrical concepts, many theoretical results were obtained in the analysis of the extension complexities of a wide variety of different geometries. In this thesis, we will mainly focus on n -gons, i.e. polygons³ with n sides.

²The goal of rigidity theory is to determine whether a given partial set of distances $d_{ij} = \|p_i - p_j\|$ between n points in $\mathbb{R}^d$ uniquely determines the coordinates of the points $p_1, \dots, p_n$ up to rigid transformations (translations, rotations, reflections) [SC10].

³A polygon is here defined as a convex and closed region of the plane enclosed by a simple cycle of straight line segments [TOG17].

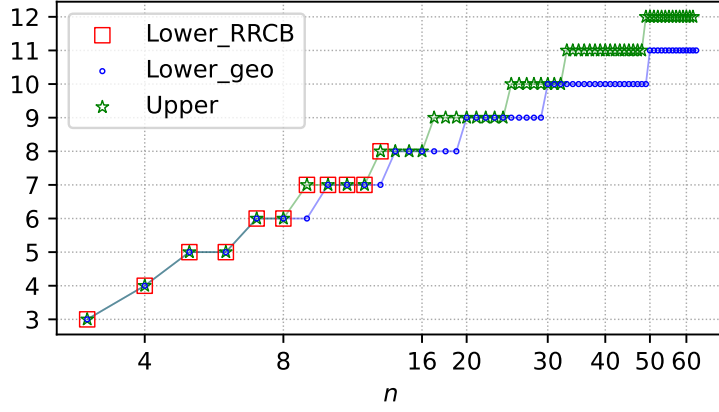


Figure 3.2 State-of-the-art lower and upper bounds for regular n -gons: *Lower_RRCB* [OVW14], *Lower_geo* [GG12b] and *Upper* [VGG17].

Regular n -gons

The most known n -gons are undoubtedly the regular n -gons. As a reminder, a *regular n -gon* is an equiangular (all angles are equal in measure) and equilateral (all sides have the same length) n -gon. A lot of research effort has been devoted to determining the extension complexity of small n -gons or equivalently the non-negative rank of their slack matrix $\mathcal{S}_n$ and how the extension complexity evolves with n . A bunch of bounds has been developed to narrow down the interval of possible values theoretically.

Lower bounds. Using a counting argument, the authors of [Goe15] showed that

$$\text{rank}_+(\mathcal{S}_n) \geq \lceil \log_2(2n + 2) \rceil. \quad (3.7)$$

Based on a refined geometric counting argument, Gillis and Glineur [GG12b] described an even stronger lower bound. For small dimensions, combinatorial bounds perform even better [OVW14; VGG17].

Upper bounds. Ben-Tal and Nemirovski [BN01] first gave an extension of the regular n -gons when n is a power of two with $2 \log_2(n) + 4$ facets. Kaibel and Pashkovich [KP11a; KP13] proposed a general construction for arbitrary n of size $2 \lceil \log_2(n) \rceil + 2$ which was improved to $2 \lceil \log_2(n) \rceil$ by Fiorini, Rothvoss and Tiwary [FRT12]. This upper bound was finally slightly strengthened in [VGG17] to

$$\text{rank}_+(\mathcal{S}_n) \leq \begin{cases} 2 \lceil \log_2(n) \rceil - 1 = 2k - 1 & \text{for } 2^{k-1} \leq n \leq 2^{k-1} + 2^{k-2} \\ 2 \lceil \log_2(n) \rceil = 2k & \text{for } 2^{k-1} + 2^{k-2} \leq n \leq 2^k \end{cases} \quad (3.8)$$

In [Van+16], the authors conjecture empirically that this upper bound is tight.

As we can observe in Figure 3.2, for $n \geq 14$, the vast majority of extension complexities remain theoretically undefined.

Generic n -gons

One might ask to which extent such a dramatic decrease in the number of facets needed to characterize a given n -gon is possible when it becomes non-regular.

A *generic n -gon* is an n -gon for which the vertices were generated randomly⁴. It can be understood that, as n increases, it becomes more and more difficult to generate generic n -gons (because it is likely that a newly generated point belongs to the convex hull of the previously generated points). Without loss of generality, we will therefore restrict the analysis to *cyclic n -gons*, i.e. random n -gons whose vertices lie on the unit circle⁵.

For a long time, the growth of the extension complexity of a given n -gon with its dimension remained unclear: is it proportional to $\sqrt{n}$, n , or something in between? [Van+16] Although it has been heavily discussed, no progress on either bound except for improving the constant factors in front $\sqrt{n}$ and n was achieved. Ultimately, Shitov came up with a very strong result in [Shi14a]: any n -gon is a linear projection of a higher-dimensional polytope with at most $147n^{2/3}$ facets. Another key contribution [KSZ22] states that the extension complexity of any n -gon $\mathcal{P}$ is likely to be about the square root of the number of vertices, i.e.

$$xc(\mathcal{P}) = \Omega(\sqrt{n}) \quad a.a.s.^6 \quad (3.9)$$

More specifically, for cyclic n -gons (with slack matrix $\hat{\mathcal{S}}_n$), they showed that it turned out that $\text{rank}_+(\hat{\mathcal{S}}_n) \leq 24\sqrt{n}$. All in all, we have that

$$\sqrt{2}\sqrt{n} \leq \text{rank}_+(\hat{\mathcal{S}}_n) \leq 24\sqrt{n} \quad (3.10)$$

where the lower bound was established in [FRT12]. Independently in [Shi14b] and [PP14], a tighter upper bound was found for small generic n -gons :

$$\text{rank}_+(\hat{\mathcal{S}}_n) \leq \left\lceil \frac{6n}{7} \right\rceil \quad (3.11)$$

Vandaele et al. [Van+16] empirically conjectured an even more restrictive upper bound:

$$\text{rank}_+(\hat{\mathcal{S}}_n) \leq \left\lceil \frac{n+6}{2} \right\rceil \quad (3.12)$$

and that this bound is tight for $5 \leq n \leq 15$.

⁴The exact definition given in [FRT12] is the following: ‘A polygon in $\mathbb{R}^2$ is generic if the coordinates of its vertices are distinct and form a set that is algebraically independent over the rationals’. It is worth mentioning that the study of random polytopes is a classical topic in probability theory [WW93].

⁵Note that we thereby use an even more general framework that in [Van+16] by not putting any constraints on the angles of the n -gon. This seems to make the problem even harder in general even for relatively small dimensions (see section 5.3.1) making it a particularly suitable setting to test the conic-based heuristics.

⁶We say that an event $\mathcal{A} = \mathcal{A}(n)$ holds asymptotically almost surely (*a.a.s.*) if $\mathbb{P}[\mathcal{A}(n)] \rightarrow 1$ as $n \rightarrow \infty$. By the asymptotic notation $xc(\mathcal{P}) = \Omega(\sqrt{n})$ we mean that there exist positive constants C and c such that $c\sqrt{n} \leq xc(\mathcal{P}) \leq C\sqrt{n}$ for all (sufficiently large) n .

3.2.3 Linear euclidean distance matrices

Over the past decade, the family of *Euclidean distance matrices*, or EDMs, have been receiving increased attention. This is mainly due to the fact that the many applications of EDMs, such as molecular conformation in bioinformatics, dimensionality reduction in machine learning and statistics, and especially the problem of wireless sensor network localization, have all become very active areas of research [KW12].

Let $\alpha = (\alpha_1, \dots, \alpha_n)$ be a real vector with $n \geq 3$. The linear Euclidean distance matrix (LEDM) of α is defined as the $n \times n$ matrix D^α whose (i, j) th entry equals $D_{ij}^\alpha = (\alpha_i - \alpha_j)^2$. By assuming $\alpha_i \neq \alpha_j \forall i \neq j$, Beasley and Laffey [BL09] show that these matrices have $\text{rank}(D^\alpha) = 3$ and that the non-negative rank of a matrix with fixed rank can be made as large as desired (while increasing the size of the matrix), implying that an upper bound for the non-negative rank of a matrix based only on the rank cannot exist. In addition, they conjectured that the non-negative rank of linear EDM's is maximum, i.e., it is equal to their dimension. First, Lin and Chu [LC10, Theorem 3.1] claimed to have proved that this equality always holds but a fatal mistake was identified in their proof. A counter-example in [GG12b], a key result on a subfamily of linear EDMs in [Hru12]

$$\text{Rank}_+(D^{1:n}) \leq 2 \log_2(n) + 2 \quad (3.13)$$

and a decisive bound in [Shi14b]

$$\text{Rank}_+(D^\alpha) \leq \left\lceil \frac{6n}{7} \right\rceil \quad \forall \alpha \in \mathbb{R}^n \quad (3.14)$$

definitively refuted the conjecture. Based on their conjecture on the non-negative rank of generic n -gons (3.12), Vandaele et al. [Van+16] empirically conjecture that

$$\text{Rank}_+(D^\alpha) \leq \left\lceil \frac{n+6}{2} \right\rceil \quad \forall \alpha \in \mathbb{R}^n. \quad (3.15)$$

3.2.4 Unique disjointness problem

The unique disjointness problem (UDISJ) is a variant of the famous set disjointness problem (DISJ) from communication complexity (see section 3.1.1 and survey [CP10]).

In DISJ, both Alice and Bob receive a subset of $[n] := \{1, 2, \dots, n\}$, and they have to determine whether the two subsets are disjoint. UDISJ is the promise version of DISJ where they have to determine whether the two subsets have zero or one element in common. Mathematically, considering the binary encoding of the sets of Alice and Bob by $a, b \in \{0, 1\}^n$, respectively, the $2^n \times 2^n$ communication matrix corresponding to this problem, referred to as the *UDISJ matrix*, has the form

$$X_n(a, b) = \begin{cases} 1 & \text{if } a^\top b = 0 \\ 0 & \text{if } a^\top b = 1 \\ \geq 0 & \text{otherwise} \end{cases} \quad (3.16)$$

It is known that the communication complexity of UDISJ is $\Omega(n)$ bits for deterministic, nondeterministic, and even randomized communication protocols [KS92; Raz92; Bar+04]. One consequence of this is that the non-negative rank of any UDISJ matrix (filled with arbitrary values in its blank entries) is $2^{\Omega(n)}$ (see equation (3.3)).

An example of a UDISJ matrix is given by $U_n(a, b) = (1 - a^\top b)^2$ where $a, b \in \{0, 1\}^n$. For $n = 3$,

$$U_3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 4 \end{bmatrix}, \quad (3.17)$$

and $\text{rank}(U_3) = 7$. None of the state-of-the-art lower bound presented in section 1.2.1 can improve the trivial bound $\text{rank}(U_3) = 7 \leq \text{rank}_+(U_3)$. It is an open problem to decide whether $\text{rank}_+(U_3) = 7$ or $\text{rank}_+(U_3) = 8$ [Gil20]. Can one prove stronger lower bounds? In particular, it has been conjectured that $\text{rank}_+(U_n) = 2^n$ in [Van+16].

Note that the interest in the UDISJ matrix exceeds the communication complexity context. In fact, it appears as a sub-matrix in many combinatorial problems and has been instrumental in proving the lower bounds of (approximate) extended formulations [BP16]. In particular, matrix U_n is a sub-matrix of the slack matrix of the correlation polytope [Fio+12].

ExactNMF via conic-based catalyser

FROM chapter 2, it is clear that the possible full-space exploration made possible by the NMU and NMO conic formulations could lead to interesting alternative methods to address the ExactNMF problem. This chapter contains the main algorithmic contributions of this thesis. We will first highlight some limitations of the existing conic-based heuristic proposed in [Lep+21] and then introduce new heuristics to explore the full potential of the full-space conic approach. In particular, we will propose some potential virtuous associations with existing state-of-the-art BCD methods.

4.1 There is a place for improvement

As it was introduced in section 2.2.2, the CFW algorithm heavily relies on the SPI procedure to provide high-quality factorizations in the most difficult cases, i.e. when the optimization process is 'stuck' due to numerical limitations of the solver.

4.1.1 "Brutality" of the SPI refinement

The SPI procedure clearly takes its roots in the sparse nature of NMF (see section 1.1.2). Starting from this intuitive consideration, it implicitly assumes that sufficiently small elements in the factors must be equal to 0, i.e. belong to the sparsity pattern. Even though it is computationally effective and easy to implement, two precautions must be taken here:

- First, the SPI threshold δ_{SPI} is insensitive to the scaling of the input matrix X . A way to counter this problem consists in normalizing successively all the columns and rows of X . This operation can be performed without deteriorating the original data and the initial scaling can easily be recovered at the end of the optimization process. Unless otherwise specified, this normalization trick will always be applied to the input data.

- Second, the SPI refinement is by construction unable to distinguish between true and false 0's. For example, situations where an exactNMF involves elements smaller than δ_{SPI} cannot be treated correctly. One could think of fixing a very small threshold to avoid the identification of false 0's. But imposing a very strict criterion could make the SPI procedure inactive or even counterproductive w.r.t. its original purpose, i.e. improve the quality of the factorization by identifying the sparsity pattern. This limitation is illustrated in Figure 4.1. It is here difficult to come up with a good trade-off for the value of δ_{SPI} which would prove satisfactory in a vast majority of cases.

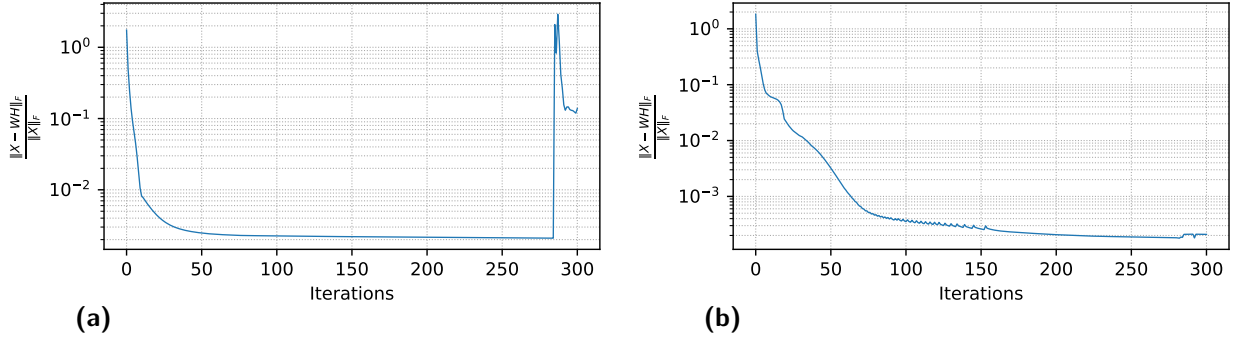


Figure 4.1 Illustration of the effect of the SPI procedure for matrix R_9^* (see section 5.2.2) and $r = 4$ with (a) $\delta_{SPI} = 10^{-4}$ (b) $\delta_{SPI} = 10^{-6}$ on the relative approximation error. In both cases, the SPI procedure (performed at iteration 280) is doing worse than better.

All in all, the SPI refinement is very arbitrary in nature and corresponds to a quite brutal (depending on the value of δ_{SPI}) ‘sparsity jump’ in the solution space with the hope that the quality of the factorization will be improved. Note that, in the rest of this thesis, we will consider $\delta_{SPI} = 10^{-6}$.

4.1.2 Towards a smooth and cautious refinement

Since the SPI procedure appears to lack sensitivity, one could wonder if another method could more nimbly identify the underlying sparsity patterns. We here directly think of the state-of-the-art BCD algorithms introduced in section 1.4.

Switching to a BCD method after the CFW algorithm applied to either NMU or NMO seems to be promising in several respects. On the one hand, these methods are specifically designed to optimally address the full $\mathbf{l}_2$ -NMF problem (not only under- or over-approximations) which is ultimately the problem on which we seek to perform. The solution space is doubled¹ which gives more degrees of freedom to find high-quality factorizations. On the other hand, these methods are progressive and monotonic² implying that they can only improve the factorizations obtained at the end of the CFW

¹In the sense that both the under- or over-approximations part of the solution space can be explored.

²Monotonicity is not theoretically ensured for all BCD methods but, as a first approximation, we will here suppose that they globally behave monotonically.

algorithm. Indeed, a conic-BCD combination inherits from the strong theoretical convergence results applying to the BCD method.

From our observations, it seems that when applying HALS at the end of the CFW optimization process, the sparsity pattern is well integrated and that, on top of this, the factors were further improved (see Figure 4.2).

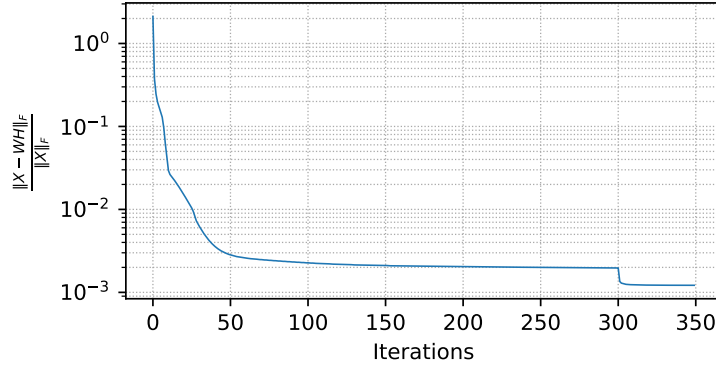


Figure 4.2 Illustration of the beneficial effect of a HALS refinement (activated at iteration 300) for matrix R_9^* (see section 5.2.2) and $r = 4$ on the relative approximation error.

These early experiments clearly indicate that the analysis of the potential complementarity between conic-based and BCD methods is an interesting direction of research.

4.2 Change of paradigm for conic NMO/NMU

As shown in the previous sections, the CFW algorithm has one very strong and promising asset: it has a great capacity for mobility through (half of) the search space. Unfortunately, this great potential comes with two less desirable proprieties. First, as it was already mentioned in section 2.2.2, its intensive and numerous calls to IPMs to solve the LMO's makes it computationally heavy and poorly scalable. In addition, even though we observe that when applying CFW the l_1 -objective being optimized and the least square approximation error evolve jointly, this 'quasi-parallel' conic-based method seems to struggle a lot to finish the job, i.e. reach the exactNMF threshold $\gamma = 10^{-6}$.

Taking all the previous considerations into account,

" This thesis will explore the potential of using the conic NMO and NMU formulations as catalyzers to perform better solution space exploration towards exact non-negative factorizations while limiting the computational weight of the IPMs as much as possible. "

In other words, given the fact that BCD method highly depends on the quality of the initialization (see section 2.1.1), we hope that the good mobility properties of the conic-based approach will result in a more robust exploitation of each initialization towards exactNMFs.

4.2.1 Limit the use of IPMs in `CFW`

Before considering the new conic-BCD combination context, let us first highlight a typical behavior of the `CFW` method which will prove to be a key element in designing the conic-BCD interaction.

Quick convergence and useless IPM calls

As could already be noted in Figures 2.1, 4.1 and 4.2, in the vast majority of cases, the mobility ability of the `CFW` methods is particularly exploited in its first iterations which in fact lead to the greatest improvements. To cut a long story short, convergence is achieved in a quite low amount of iterations (compared to BCD methods) and especially in the first ones.

Progressive SPI refinement [pSPI]

Given this interesting convergence property, a first area of improvement can be identified in the conic context. Instead of applying the SPI refinement at the end of the `CFW` algorithm as proposed in [Lep+21], we could think of performing the ‘sparsity jumps’ earlier in the optimization process in order to avoid superfluous iterations. In this way, the sparsity pattern gets integrated little by little³ making the SPI procedure more progressive. We call this new refinement procedure *progressive SPI refinement* which will be denoted pSPI.

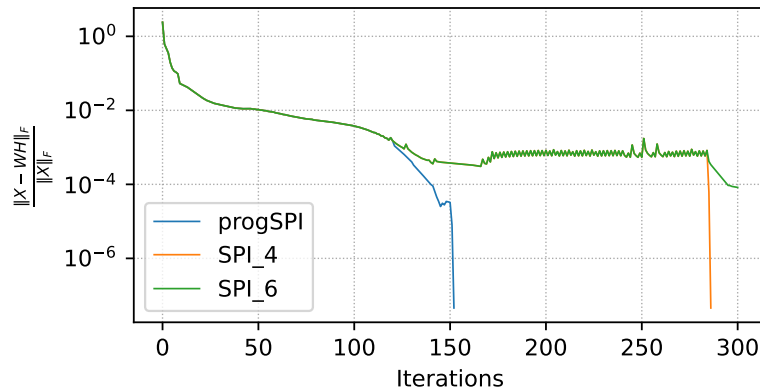


Figure 4.3 Illustration on matrix M_3 (see section 5.2.1) with $r = 4$ of the convergence benefits of the progSPI refinement (activated at iterations 120, 132, 141 and 151) compared with the SPI procedure (activated at iteration 285) with $\delta_{SPI} = 10^{-4}$ and $\delta_{SPI} = 10^{-6}$.

The potential convergence benefits⁴ of this new refinement technique is illustrated at Figure 4.3. It appears that the different ‘sparsity jumps’ can have a more or less significant impact on the relative approximation error but they all seem to be decisive in the convergence process. Note that the maximum number of iterations of the methods was here fixed to 300. The gain in iterations would have been even greater if we

³Once a stagnation of the relative approximation error is observed (see section 4.2.2), every 10 iterations, the SPI refinement is applied on the current factors.

⁴Even if in this example progSPI clearly outperforms SPI, there are no guarantees that in general progSPI leads to better factorizations than SPI.

had set this parameter to a higher value. We here understand that implementing a convergence-aware refinement can lead to important iteration savings.

Note that, as a complement to the discussion of section 4.1.1 concerning the limitations of the SPI refinement procedure, in Figure 4.3, we can observe a case where a too strict SPI threshold prevents the identification of a proper sparsity pattern. Finally, it is worth mentioning that progSPI has in general the same potential limitations as SPI.

4.2.2 Convergence-aware conic-BCD switches

Building on the observations made in the previous section, with the aim of keeping the full optimization process as scalable and effective as possible, we would like to limit the use of expensive full-space iterations whenever possible. So, in addition to its final imprecision, the computational price to pay for each iteration of the CFW algorithm is certainly also a strong incentive to consider it as an 'exactNMF catalyzer'. It becomes clear that a good exploration trade-off must here be designed :

When should we switch from the expensive full-space CFW algorithm to a cheap BCD refinement method ?

Several complementary strategies can be devised to accomplish the mission of reducing as much as possible the IPM calls while preserving the interesting mobility properties.

Convergence switch: Plateau

As discussed in the previous sections, it seems to make sense to try to

"monitor the convergence of the CFW algorithm to avoid unnecessary expensive iterations."

A natural way to achieve this is to identify a stagnating behavior of the objective function. We here define a *plateau* as a succession of $\alpha = 20$ iterations where the relative improvement of the l_1 -objective function, which is defined at iteration i as

$$e_1(i) = \left| \frac{\sum_{f,g,k} [W_{i-1}(f,k)H_{i-1}(k,g) - W_i(f,k)H_i(k,g)]}{\sum_{f,g,k} W_i(f,k)H_i(k,g)} \right|, \quad (4.1)$$

has not achieved sufficient progress, i.e. $e_1(i) < \mu$. We set the *plateau threshold* at $\mu = 10^{-4}$ which is not too severe. The reason for this is that a refinement follows anyway and it is therefore useless to persist too long when a relative stagnation is observed.

Region of interest switch: Threshold

It is possible to consider the conic formulations even more as exactNMF catalyzers. In fact, when the CFW algorithm has moved from a given initialization to an interesting region of the solution space, we could think of switching to a BCD method before convergence is observed to explore that region faster. Here, the rationale behind the switching mechanism is that

"the role of the CFW algorithm may be limited to the identification of a promising region with respect to the ExactNMF problem."

More concretely, this mechanism can be implemented by a simple *threshold*. We will consider that the CFW method has done the job when it has discovered a basin of attraction with at least $\|X - WH\|_F / \|X\|_F \leq \sigma$ where we arbitrarily⁵ fix $\sigma = 10^{-3}$.

As Figure 4.4 suggests the above strategies can in some cases lead to a more fruitful or sparing exploration of the solution space. But there is, to the best of our knowledge, no reasonable explanation by which one would expect a better or worse exploration in general. Their only unquestionable benefit is their important savings of IPM calls.

Of course, for some difficult cases, the threshold will not be attained showing the potential complementarity between both criteria.

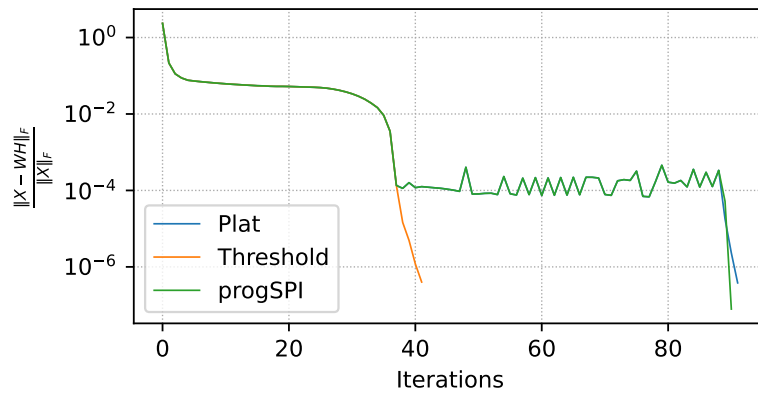


Figure 4.4 Illustration on matrix M_3 (see section 5.2.1) and $r = 4$ of the *plateau* and *threshold* switches along with the progSPI refinement.

4.2.3 Switching back to CFW?

So far, we have considered the BCD algorithms as refinement methods to finish to job started by the CFW algorithm. Based on the observation that the CFW algorithm usually gets stuck at a factorization that can still be improved w.r.t. its objective function (see section 2.2.2), one could think of the BCD methods as a helping boost to continue the conic-based optimization process and thereby come up with a method implementing the following sequence: CFW-BCD-CFW-BCD.

However, in practice, this idea has proved to be quite unpromising. There are two main reasons to explain this lack of success. The first obvious explanation is that both approaches are not constrained by the same domain. In a lot of situations, BCD methods exploit their full solution space by navigating through both the under- and overapproximation domains. If the "BCD-boost" provides a solution in the other sub-domain,

⁵This arbitrary value was chosen based on the observation that when the switch is operated at that stage, in a vast majority of cases, this leads to a success. In practice, $\sigma = 10^{-3}$ appears as a good trade-off between the complete exploitation of the conic-based formulation and the BCD refinement.

it usually appears as a quite poor re-initialization of the [CFW](#) method. In addition, even if we stay in the same sub-domain during the "BCD-boost", another problem is usually encountered. Since the BCD methods seek to optimize a different objective function, when switching back to the [CFW](#) algorithm, we often observe that the method tends to recover the factorization where it was stuck.

All in all, the "BCD-boost" can be considered superfluous in most of the practical situations.

4.3 Conic-based heuristics for [ExactNMF](#)

Based on the switch strategies proposed in section 4.2.2 and on the already successful heuristic approaches proposed in [[Van+16](#)], this section takes up the main contribution of this thesis, namely new conic-based heuristics to address the [ExactNMF](#) problem.

Before diving into the design of the proposed methods, to clearly understand their positionality, it seems of utmost importance to clarify their purpose properly.

The goal of the new conic-based heuristics is to compute exactNMFs where the existing heuristics cannot, whatever the computational load.

Of course, the algorithms must remain within an acceptable computing reach, hence the switches developed in section 4.2.2. As suggested by previous developments on the poor scalability of the conic approach, their aim is therefore surely not to outperform the state-of-the-art methods in speed for "easy cases" and a fortiori not even to improve success rates for known factorizations either.

4.3.1 "Fully initialized" heuristics [C, C_pSPI, C_TP, C_TMBS]

The first bunch of heuristics includes only methods that are *fully initialized*, in the sense that the initializations never depend on the previous states of the iterates. Building on previous developments, they implement increasingly sophisticated mechanisms to incorporate the conic-BCD switches in the most promising way.

[CFW](#) with final BCD refinement [C]

The first heuristic stems from the discussion around the 'brutality' of the SPI refinement procedure (see section 4.1.2) and is the direct counterpart of the C_SPI heuristic, i.e. the [CFW](#) algorithm with a final SPI refinement. The final SPI refinement is simply replaced by a BCD refinement (see Algorithm 10).

[CFW](#) with progSPI and final BCD refinement [C_pSPI]

The second heuristic is an attempt to conciliate the benefits of both refinement paradigms, i.e. the progSPI (implementing a [plateau](#) mechanism) and the BCD refinement, which are in theory not incompatible and may even lead to an interesting combination in

practice. This method aims at exploiting the conic-based formulation in a more complete manner, i.e. including the potentially beneficial ‘sparsity jumps’, before switching to the final BCD refinement (see Algorithm 11).

Algorithm 10. *CFW* with final BCD refinement [C]

Input : $X \in \mathbb{R}_+^{m \times n}$, r and init type $\mathcal{I}$.

Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

- 1: $Z^{(0)} \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$
- 2: **for** $i = 1, \dots, \text{maxiter}$ **do**
- 3: Compute $Z^{(i)} = \arg \min_{Z \in Q} \langle \nabla \Phi(Z^{(i-1)}), Z \rangle$ [LMO]
- 4: **end for**
- 5: $[W, H] \leftarrow \text{BCD_Refinement}(X, Z^{(end)})$

Algorithm 11. *CFW* with progSPI and final BCD refinement [C_pSPI]

Input : $X \in \mathbb{R}_+^{m \times n}$, r and init type $\mathcal{I}$.

Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

- 1: $Z^{(0)} \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$
- 2: **for** $i = 1, \dots, \text{maxiter}$ **do**
- 3: Compute $Z^{(i)} = \arg \min_{Z \in Q} \langle \nabla \Phi(Z^{(i-1)}), Z \rangle$ [LMO]
- 4: **if** $\text{Plateau}(Z^{(0:i)})$ **then**
- 5: $Z^{(i)} \leftarrow \text{prog_SPI}(Z^{(i)})$
- 6: **end if**
- 7: **end for**
- 8: $[W, H] \leftarrow \text{BCD_Refinement}(X, Z^{(end)})$

CFW with threshold and plateau BCD refinement [C_TP]

In the following heuristic and from now on, we get totally rid of any SPI refinement procedure and use the **plateau** identification strategy as a strict conic-BCD switch. For the same initialization of the factors, this method will therefore involve way fewer IPM calls. On top of that, with, once again, the concern to reduce as much as possible the IPM calls, we will here consider a global **threshold** switch. This means that, when a factorization with $\|X - WH\|_F / \|X\|_F = \sigma$ is found, the final BCD refinement is automatically launched (see Algorithm 12).

CFW with threshold multi-best-start and plateau BCD refinement [C_TMBS]

Even if we have identified an interesting mobility potential for the *CFW* algorithm (see section 2.2.2) and thereby may expect better initialization exploitation in general, since

Algorithm 12. CFW with threshold and plateau BCD refinement [C_TP]**Input :** $X \in \mathbb{R}_+^{m \times n}$, r and init type $\mathcal{I}$.**Output :** $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $Z^{(0)} \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
2: for  $i = 1, \dots, \text{maxiter}$  do
3:   Compute  $Z^{(i)} = \arg \min_{Z \in Q} \langle \nabla \Phi(Z^{(i-1)}), Z \rangle$  [LMO]
4:   if  $\text{Plateau}(Z^{(0:i)})$  or  $\text{Threshold}(Z^{(i)})$  then
5:     break
6:   end if
7: end for
8:  $[W, H] \leftarrow \text{BCD\_Refinement}(X, Z^{(end)})$ 

```

we aim at solving the most difficult highly non-convex problems, it seems still relevant to design heuristics that explore the solution space from different starting points, i.e. *multi-start methods*. The different mobilized initialization will be specified in section 4.3.4.

As with the MBS heuristic, it makes sense to keep only the best start to continue the optimization process, especially since we here expect that the main progress is made in the first iterations of the CFW algorithm. Each of the K initializations gets N LMO iterations to reveal its potential. In addition, as introduced in the previous heuristic, we will here again consider a global threshold switch. Note that in the second phase (with the most promising start) of the CFW algorithm a plateau identification strategy is involved as a strict conic-BCD switch (see Algorithm 13).

Algorithm 13. CFW with threshold MBS and plateau BCD refinement [C_TMBS]**Input :** $X \in \mathbb{R}_+^{m \times n}$, r , N and init types $\mathcal{I}_1, \dots, \mathcal{I}_K$.**Output :** $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $Z^{(0:\tilde{N})} \leftarrow \text{Multi\_Best\_Start}(\mathcal{I}_1, \dots, \mathcal{I}_K, N, m, n, r)$ 
2: for  $i = \tilde{N}, \dots, \text{maxiter}$  do
3:   Compute  $Z^{(i)} = \arg \min_{Z \in Q} \langle \nabla \Phi(Z^{(i-1)}), Z \rangle$  [LMO]
4:   if  $\text{Plateau}(Z^{(0:i)})$  or  $\text{Threshold}(Z^{(i)})$  then
5:     break
6:   end if
7: end for
8:  $[W, H] \leftarrow \text{BCD\_Refinement}(X, Z^{(end)})$ 

```

4.3.2 "Partially initialized" heuristics [C_RBR, C_TSA, C_HYB]

The second class of heuristics includes methods that can be identified as *partially initialized*, in the sense that some starting points depend on the previous states of the iterates. In other words, the new starting factors are composed of parts of a previous factorization and newly initialized elements.

Rank-by-rank CFW with threshold and plateau BCD refinement [C_RBR]

Following the constructive paradigm proposed in RBR, we introduce an equivalent *rank-by-rank* heuristic for the conic-based approach. Starting from an optimal rank-1 solution (see section 1.2.2), the other dimensions are added to the optimization process one at a time. For each of them, K attempts with N LMO iterations are performed to find the optimal approximation of a given rank before augmenting the factorization rank (see Algorithm 14).

Algorithm 14. RBR CFW with threshold and plateau BCD refinement [C_RBR]

Input : $X \in \mathbb{R}_+^{m \times n}$, r , K , N .

Output : $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $[u_1, \sigma_1, v_1] \leftarrow \text{SVD}(X)$ 
2:  $(W_1, H_1) \leftarrow (\sqrt{\sigma_1}|u_1|, \sqrt{\sigma_1}|v_1|)$  % See section 1.2.2
3: for  $k = 2, \dots, r$  do % Add one dimension at the time.
4:    $e_{\min} \leftarrow 1$ 
5:   for  $i = 1, \dots, K$  do
6:      $(\tilde{W}_k(:, 1:k-1), \tilde{H}_k(1:k-1,:)) \leftarrow (W_{k-1}, H_{k-1})$ 
7:      $(\tilde{W}_k(:, k), \tilde{H}_k(k, :)) \leftarrow \text{Rand\_Initialization}(m, n, 1)$ 
8:      $(\tilde{W}_k, \tilde{H}_k) \leftarrow \text{Plateau\_LMO}(X, \tilde{W}_k, \tilde{H}_k, N)$ 
9:      $\tilde{e} \leftarrow \|X - \tilde{W}_k \tilde{H}_k\|_F / \|X\|_F$ 
10:    if  $\tilde{e} < e_{\min}$  then
11:       $(W_k, H_k) \leftarrow (\tilde{W}_k, \tilde{H}_k)$ ,  $e_{\min} \leftarrow \tilde{e}$ 
12:      if  $\text{Threshold}(e_{\min})$  then
13:        break
14:      end if
15:    end if
16:  end for
17: end for
18:  $[W, H] \leftarrow \text{BCD\_Refinement}(X, W_r, H_r)$ 

```

Again, conic-BCD switches are incorporated into the method. On the one hand, there is a global **threshold** over the complete CFW run. And on the other hand, a **plateau** identification mechanism is implemented for each dimension leading to potentially large computational savings (see Figure 4.5). Note that the *plateau threshold* and the

plateau counter are respectively reduced to $\mu = 10^{-2}$ and $\alpha = 5$ for all the intermediate dimensions to avoid unnecessary IPM calls and thereby obtain an optimized cadence between the dimensions.

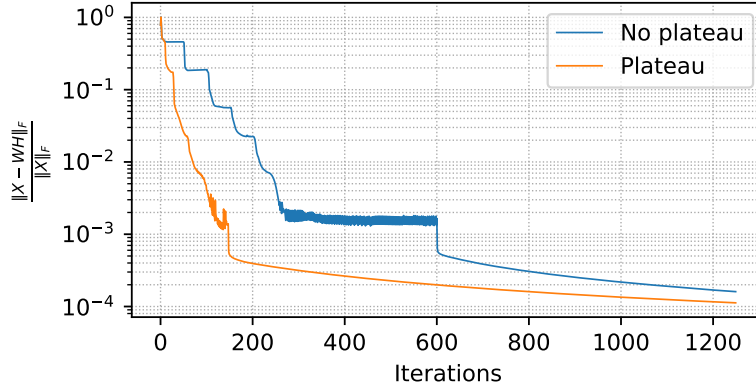


Figure 4.5 Illustration on matrix $D^{1:8}$ (see section 3.2.3) with $r = 6$ of the *plateau* identification mechanism in the `C_RBR` heuristic.

This heuristic can be seen as a *partially initialized multi-best-start* method, in the sense that when adding a dimension, K different initializations of that new dimension are considered and only the best pair of resulting factors are kept. This constructive multi-start approach has also some potential computational benefits. In fact, while optimizing the first dimensions, the global size of the problem is smaller which will lead to faster `LMO` iterations.

Threshold simulated annealing CFW with final BCD refinement [C_TSA]

Another partially initialized method is inspired by the *simulated annealing* exploration strategy implemented in the `SA` heuristic. As already described in section 2.1.3, this local exploration is performed with P decreasing acceptance levels (given by temperatures between T_0 and T_{end}) with K attempts for each level. The best-identified solution is then finally refined to obtain an exact factorization (see Algorithm 15)⁶.

The major difference with the `SA` heuristic is that, with the aim of using the conic-based approach as an exactNMF catalyzer, we add a global *threshold* during the exploration process to switch to a BCD refinement as soon as a sufficiently promising region of the solution space is identified. Nevertheless, this time, if an exactNMF is not found after the *threshold* switch, the optimization process goes back to the conic-based simulated annealing exploration.

Hybridization of [C_HYB]

Finally, since a hybridization of the `RBR` and `SA` has led to great results in [Van+16], we also consider a combination of their conic counterparts, i.e. `C_RBR` followed by `C_TSA`.

⁶Please refer to section 2.1 for the definition of the different hyperparameters of Algorithm 15.

Algorithm 15. Simulated annealing heuristic [SA]**Input :** $X \in \mathbb{R}_+^{m \times n}$, $r, K, N, J, \gamma, T_0, T_{end}, P$ and init type $\mathcal{I}$.**Output :** $(W, H) \geq 0$: A rank- r ℓ_2 -NMF of $X \approx WH$.

```

1:  $[W_\#, H_\#] \leftarrow \text{Initialization}(\mathcal{I}, m, n, r)$ 
2:  $e \leftarrow \|X - W_\# H_\#\|_F / \|X\|_F$ ,  $e_{min} \leftarrow e$ 
3:  $T \leftarrow T_0$ 
4: while  $T > T_{end}$  do
5:   for  $i = 1, \dots, K$  do
6:      $(\tilde{W}_\#, \tilde{H}_\#) \leftarrow (W_\#, H_\#)$ 
7:      $\mathcal{J} \leftarrow$  pick randomly  $J$  indices in  $\{1, \dots, r\}$ 
8:      $[\tilde{W}_\#(:, \mathcal{J}), \tilde{H}_\#(\mathcal{J}, :)] \leftarrow \text{Rand\_Initialization}(m, n, J)$ 
9:      $[\tilde{W}_\#, \tilde{H}_\#] \leftarrow \text{Plateau\_LMO}(X, \tilde{W}_\#, \tilde{H}_\#, N)$ 
10:     $\tilde{e} \leftarrow \|X - \tilde{W}_\# \tilde{H}_\#\|_F / \|X\|_F$ 
11:    if  $\text{Threshold}(\tilde{e})$  then
12:       $[W^*, H^*] \leftarrow \text{BCD\_refinement}(X, \tilde{W}_\#, \tilde{H}_\#)$ 
13:      if  $\|X - W^* H^*\|_F / \|X\|_F < \gamma$  then
14:         $(W, H) \leftarrow (W^*, H^*)$ 
15:         $T = T_{end}$ , break
16:    end if
17:    else if  $\tilde{e} < e$  or  $U[0, 1] < \exp(\frac{e - \tilde{e}}{T})$  then
18:       $(W_\#, H_\#) \leftarrow (\tilde{W}_\#, \tilde{H}_\#)$ ,  $e \leftarrow \tilde{e}$ 
19:      if  $e < e_{min}$  then
20:         $(W, H) \leftarrow (W_\#, H_\#)$ ,  $e_{min} \leftarrow e$ 
21:        if  $e_{min} < \gamma$  then
22:           $T = T_{end}$ , break
23:        end if
24:      end if
25:    end if
26:  end for
27:   $T \leftarrow T - (T_0 - T_{end}) / P$ 
28: end while
29:  $[W, H] \leftarrow \text{BCD\_refinement}(X, W, H)$ 

```

4.3.3 Simultaneity : Necessary condition and great asset

It is here important to note that in the previous heuristics, all the variables are always simultaneously involved in the LMOs. Why is this so? As already discussed in section 2.2.2, a block approach seems inappropriate in the context of the conic-based approach. Let us introduce some complementary comments to clarify even more this variable simultaneity necessity.

Let us first recall that the conic [NMO](#) and [NMU](#) problems were introduced as convex alternatives to the ℓ_2 -NMF problem. The hope was that they would show "parallel" behaviors in the sense that optimizing the factors according to one formulation would benefit the other as well. As was already the case in section 2.2.2 with the unfruitful *analytic iterative deflation process*, unfortunately, considering even bigger blocks with a [CFW](#)-like approach does not turn out to be any more conclusive. This is bad news since it could have been a great option to decrease the computational load of each IPM call.

The main reason for this impossibility lies in the crucial difference in their objective function. A block approach is very relevant to the ℓ_2 -NMF problem since the objective is an approximation error and therefore there is a strong interdependence of the different components of the factors in the objective function. All variables share the common objective to jointly and severally minimize the least square approximation error. On the contrary, in the [NMO](#) and [NMU](#) settings, the only ties between the different variables lie in the under- or overapproximation constraint. The different blocks are here not guided towards the common error minimization objective and tend to selfishly over-optimize their ℓ_1 -norm. This behavior leads to an unalignment of the conic approach with respect to the final error minimization goal.

In order to balance the 'selfishness' of all the elements and to enable optimization trade-offs between the blocks, it seems essential to take all variables into consideration in the optimization process.

To illustrate this, we have implemented a *cyclic rank-by-rank CFW* which is equivalent to the [CFW](#) algorithm apart from the fact that, in a cyclic manner, for each [LMO](#) iteration a different rank-1 contribution is optimized keeping all the others intact. In Figure 4.6, we can observe that this approach (w.l.o.g. for the [NMO](#) case) is very initialization-dependent and that, in the end, it performs as poorly from all starting factorizations with respect to the [CFW](#) algorithm.

On the one side, for heavy elementwise initialization⁷ (in dark), since the initialization leads to an exaggerated overapproximation, the first rank-1 contributions are all put to zero⁸ leaving only very few contributions out of the sparsity pattern. The sparsity pattern is here integrated in a very arbitrary and unwise manner. The independent consideration of the rank-1 blocks has the undesirable effect of automatically eliminating some key contributions.

On the other side, for light elementwise initialization (in light), since the initialization leads to an exaggerated underapproximation, the first rank-1 contribution is used to re-establish the overapproximation constraint, and the other blocks are not significantly modified afterward.

⁷In other words, initialization factors with relatively big elements compared to the normalized input matrix.

⁸This is confirmed by the plateaus observed in the evolution of the relative approximation error. Indeed, since the last rank-1 contributions were optimized to respect the overapproximation constraint, the zero-valued "deflated" contributions remain unchanged.

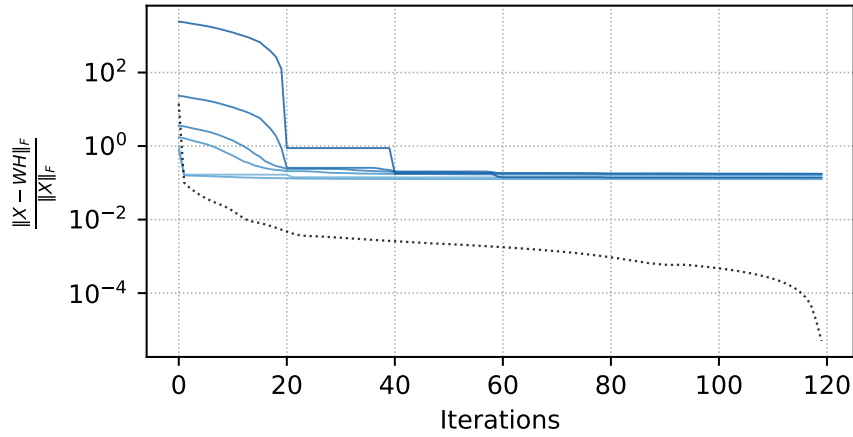


Figure 4.6 Performance comparison between the *cyclic rank-by-rank CFW* and classic *CFW* algorithm with a 20×20 random generated matrix X with $\text{rank}_+(X) = 20$. For the block approach, the spectrum of blues indicates the elementwise weight of the initialization factors: the darker the heavier. The dotted line stands for the evolution of the classic *CFW* algorithm.

All in all, the simultaneity of the variables is a necessity to ensure the behavioral alignment of the conic formulations. But, of course, a priori, it also appears as a potentially great asset of the conic approach making it a particularly promising agile and mobile paradigm (see section 2.2.2).

4.3.4 Initialization of the conic-based heuristics

As already discussed in section 1.4.7, the choice of initial factors significantly affects the convergence speed of the algorithms and the quality of the final factorization to which they converge. Following [Lep+21], for all the conic-based heuristics, we chose to always randomly initialize each entry of W and H using the uniform distribution in the interval $[0, 1]$. In general, dense random initializations have proven to work better than deterministic initializations or even sparse random initializations (which are used for BCD-based heuristics). This can be explained by the fact that dense initializations are located more centrally in the solution space, i.e. far away from the boundaries. In fact, in the context of a less "optimal" but more mobile exploration, central positions turn out to be more fruitful starting points.

Only the C_TBMS heuristic is an exception in terms of initialization. In order to ensure a wide diversity of angles of attack, 7 different random initializations (5 dense and 2 sparse), 1 'semi-random' (rank1_NMO [Lep+21]) and 3 deterministic (FastSepNMF [GV13a], hierclust2nmf [GKP14] and NNSVDLRC [AQG19]) are involved in the multi-best-start process.

It is also worth noting here that the normalization trick introduced in section 4.1.1 is also relevant with respect to the scaling of the random initialization.

Numerical Experiments

IN this last section, we will make clear through concrete numerical experiments how the conic-based heuristics introduced in this thesis can positively complement the existing BCD-based heuristics. Note that, for the sake of simplicity, we exclusively focused on the conic **NMO** formulation since it was identified as the most promising one in [Lep+21].

5.1 Towards scalability : Unlock higher dimensional problems

Let us first try to shed some light on the scalability of the previously designed methods. In fact, as mentioned before, even if we designed smart switches to reduce as much as possible the expensive IPM calls (see section 4.2.2), we still expect the conic-based heuristics to scale poorly. Let us take a quantitative look at this key scalability issue.

5.1.1 Construction of the conic optimization problem

From the first numerical experiments of the **CFW** algorithm on various **ExactNMF** problems in [Lep+21], it appeared that square matrices with dimension greater than 15 and non-negative factorization rank greater than 8 were out of reach. This was due to a very expensive construction of the conic optimization problems¹. Some significant coding efforts have been made to make higher dimensional problems within acceptable computational reach (see Figure 5.1).

5.1.2 Making **LMO** iterations cheaper?

As repeated many times before, the poor scalability of the conic-based approach comes from the important computational effort that is needed to perform each IPM-based **LMO**

¹Note that the major part of the problem formulation remains unchanged along the different **LMO** iterations of the **CFW** algorithm. Only the coefficients of the linear objective function (see section 2.2.2) are updated from iteration to iteration.

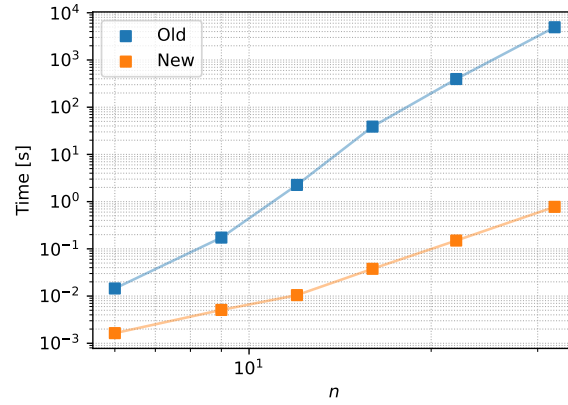
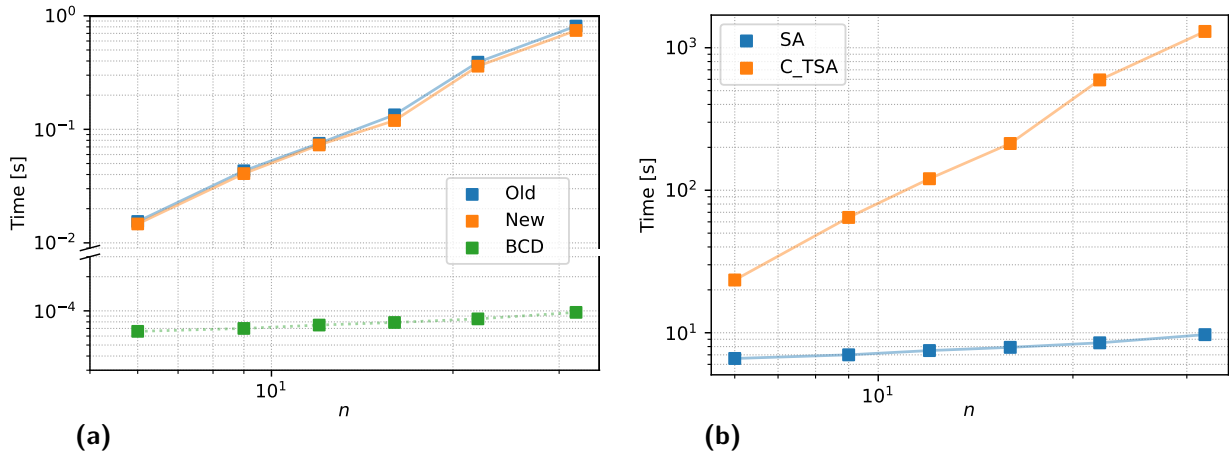


Figure 5.1 Illustration for regular n -gons ($\mathcal{S}_n$, see section 3.2.2) of increasing size of the time gain for the construction of the conic optimization problem

iteration. One can observe in Figure 5.2a that several orders of magnitude separate the execution time of the LMO iteration with the BCD iteration. One could rightly argue that this is partly due to the fact that a single LMO iteration performs a greater step towards optimality compared to a BCD iteration. But, when we take this aspect into account and consider (in Figure 5.2b) the average worst-case time for a conic-based heuristic and its BCD counterparts (which involves way more "partial" re-initializations), we observe that the evolution of the computing time with the dimension of the problem is still 400 times greater for the conic version.



IPM modifications

We first tried to modify the IPM internally by putting a limit on the number of IPM iterations. This proved to be very inconclusive leading often the CFW algorithm to diverge. This can be at least partly explained by the fact that IPMs achieve their most significant steps towards optimality in their last iterations. Note that, since we used the "black box" MOSEK software in this thesis, we have had relatively little room for manoeuvre to tune the IPM to our NMF setting. This could be a further direction of research.

Momentum schemes

Recently, it has been highlighted that the relatively slow local convergence property which constitutes a major limitation of the Frank-Wolfe algorithm is due to a zig-zagging behavior [CLS21]. This zig-zagging phenomenon can be viewed as an artifact of discretization and leads to bad step directions even arbitrarily close to the solution.

To overcome this problem, several momentum schemes both on the iterates and on the directions of search have been proposed lately [Li+20; Li+21; CS22]. They appeared to be fruitful in some specific settings but were identified as unlikely to be effective on general problems.

'Iterate' momentum. In our NMF setting, the interest of a momentum mechanism of the form

$$Z^{(i)} = Z^{(i-1)} + \tau [S^{(i)} - Z^{(i-1)}] \quad (5.1)$$

where $S^{(i)}$ is the solution of the LMO sub-problem at iteration i , remains unclear (see Figure 5.3). Faced with this ambiguous situation, we decided to keep a constant $\tau = 1$, i.e. the full mobility setting. In fact, it seems that maintaining iterates as a convex combination of a few extreme points is not needed and that it is sufficient (or better?) to keep only the last one.

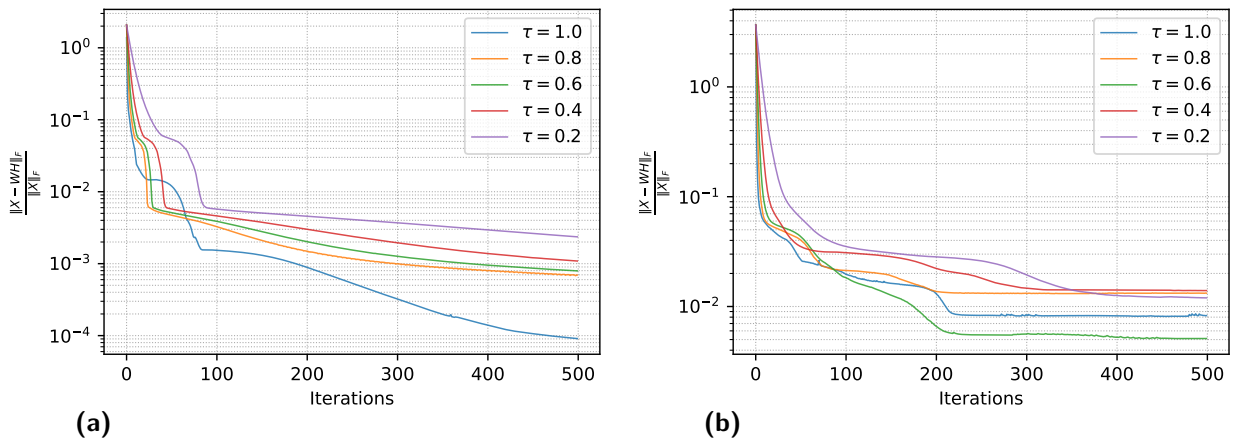


Figure 5.3 Illustration of the effect of the iterate momentum on (a) matrix R_3^* (see section 5.2.2) with $r = 4$ and (b) matrix $D^{1:8}$ (see section 3.2.3) with $r = 6$. In the first case, $\tau = 1$ seems to provide the best exploration capabilities. This is not confirmed in the second setting. Nevertheless, in both cases, a too-small τ seems to be causing a slowdown.

Direction momentum. A momentum mechanism of the form

$$d^{(i)} = d^{(i-1)} + p \left[\nabla \Phi(Z^{(i-1)}) - d^{(i-1)} \right] \quad (5.2)$$

where $d^{(i)}$ is the "linear" direction of search of the LMO sub-problem at iteration i , seems to have a slowdown effect (see Figure 5.4). Again, since we use the CFW algorithm as a catalyzer and since we seek to reduce as much as possible the IPM calls, we decide to keep a full update of the direction of search at each iteration.

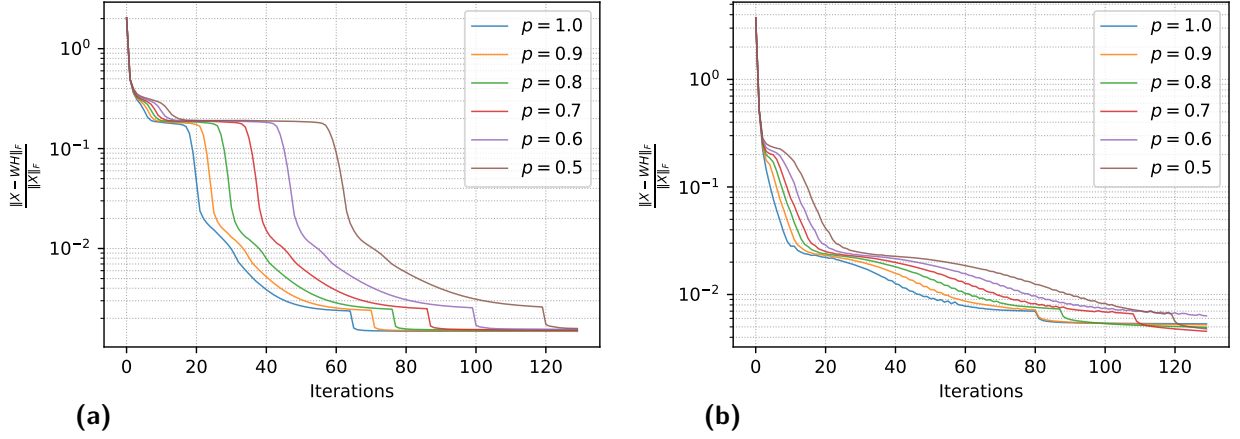


Figure 5.4 Illustration of the effect of a direction momentum on (a) matrix R_3^* (see section 5.2.2) with $r = 4$ and (b) matrix $D^{1:8}$ (see section 3.2.3) with $r = 6$. matrix 8. In both cases, a slowdown is observed.

5.2 Numerical validation experiments

The first numerical experiments presented in this section aim at confirming the value of the conic-based heuristics compared to their BCD counterparts and show the great progress achieved both in time and accuracy with respect to the early experiments conducted in [Lep+21]. To do so, we will analyze how the heuristics behave in settings of medium difficulty.

For all the BCD-based heuristics, we use the hyperparameters recommended in [Van+16]. For the conic-based ones, a good trade-off between time and exploration brought us to arbitrarily fix the hyperparameters to the following values : $\text{maxiter} = 750$ (C_SPI , C , C_pSPI , C_TP , C_TMBS), $N = 400$, $K = 5$ (C_RBR), $N = 50$, $K = 10$, $J = 2$, $T_0 = 10^{-1}$, $T_{\text{end}} = 10^{-4}$, $P = 5$ (C_TSA). All numerical experiments were performed using Matlab on a computer Intel Xeon W-2295 CPU @ 3.00GHz 2X64Go RAM.

5.2.1 Nested hexagons

To begin with, let us consider the nested hexagons problem which is a computational geometry problem frequently regarded in the literature as an archetypal example [Gil17;

MSV03; Lep+21]. Let $a > 1$ and let us consider the matrix

$$M_a = \frac{1}{a} \begin{bmatrix} 1 & a & 2a-1 & 2a-1 & a & 1 \\ 1 & 1 & a & 2a-1 & 2a-1 & a \\ a & 1 & 1 & a & 2a-1 & 2a-1 \\ 2a-1 & a & 1 & 1 & a & 2a-1 \\ 2a-1 & 2a-1 & a & 1 & 1 & a \\ a & 2a-1 & 2a-1 & a & 1 & 1 \end{bmatrix}. \quad (5.3)$$

The **ExactNMF** problem involving matrix M_a amounts to find the polytope with the smallest number of vertices whose projection in the plane lies in between two hexagons. The inner hexagon is smaller than the outer one with a ratio of $\frac{a-1}{a}$. For $a = 2$, the inner hexagon is twice as small as the outer one, and we can fit a triangle between the two and so $\text{rank}_+(M_2) = 3$ (see Figure 5.5a). For any $a > 2$, $\text{rank}_+(M_a) \geq 4$ because no triangle can fit between the two hexagons. For $a = 3$, the inner hexagon is $2/3$ smaller than the outer one, and we can fit a rectangle between the two and $\text{rank}_+(M_3) = 4$ (see Figure 5.5b). This implies that $\text{rank}_+(M_a) = 4$ for all $2 < a \leq 3$. For any $a > 3$, $\text{rank}_+(M_a) = 5$. Surprisingly, the non-negative rank of M_a is always no more than 5 (even when a tends to infinity, in which case the inner and outer hexagons coincide) because there exists a three-dimensional polytope within Δ^6 with 5 vertices that contains the outer polytope.

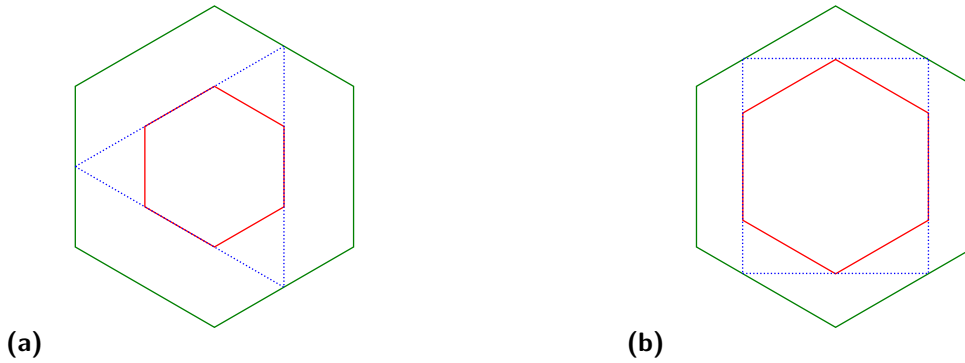


Figure 5.5 Illustration of the nested polygons problem for (a) $a = 2$ and (b) $a = 3$.

Table 5.1 reports the performance of all the **ExactNMF** heuristics presented in this thesis on matrices M_2, M_3, M_4 and M_∞ .

5.2.2 Infinitesimally rigid non-negative factorizations

Let us now analyze in Table 5.2 the performances of the complete set of heuristics on the even trickier matrices proposed in [KK21] and introduced in section 3.2.1. The numbering of the matrices corresponds to their order in the list provided in the article. Note that they are all tested with factorization rank $r = 4$.

The numerical experiments tend to confirm that the 15 infinitesimally rigid non-negative factorizations proposed in [KK21] are in fact globally rigid.

	M_2	M_3	M_4	M_∞
C_SPI	50/50 (4.76)	50/50 (5.93)	25/50 (7.73)	16/50 (8.06)
C	50/50 (4.98)	50/50 (6.28)	31/50 (8.03)	16/50 (8.56)
C_TP	50/50 (0.46)	50/50 (0.98)	21/50 (1.14)	14/50 (1.69)
C_pSPI	50/50 (1.21)	50/50 (1.62)	49/50 (1.41)	42/50 (2.44)
C_TMBS	50/50 (0.56)	50/50 (1.20)	50/50 (2.45)	48/50 (6.85)
C_RBR	50/50 (0.43)	50/50 (0.64)	47/50 (0.35)	45/50 (0.22)
C_TSA	50/50 (0.43)	50/50 (1.96)	49/50 (1.42)	49/50 (0.46)
C_HYB	50/50 (0.77)	50/50 (1.18)	50/50 (1.41)	50/50 (1.27)
MS	50/50 (0.02)	50/50 (0.19)	20/50 (0.64)	5/50 (0.01)
MBS	50/50 (0.31)	50/50 (0.46)	50/50 (0.39)	50/50 (0.34)
RBR	50/50 (0.08)	50/50 (0.31)	50/50 (0.19)	48/50 (0.17)
SA	50/50 (0.07)	50/50 (0.52)	50/50 (0.09)	50/50 (0.21)
HYB	50/50 (0.44)	50/50 (0.82)	50/50 (0.34)	50/50 (0.16)

Table 5.1 Performance comparison of the different heuristics on 4 instances ($a = 2, 3, 4, \infty$) of the nested hexagons problem exposed in section 5.2.1. The horizontal dotted line separates the heuristics of this thesis with the heuristics of [Van+16].

	R_2	R_3^*	R_5	R_9^*
C_SPI	0/50 (44.6)	0/50 (47.1)	0/50 (24.5)	0/50 (58.0)
C	3/50 (46.2)	5/50 (58.9)	13/50 (27.2)	22/50 (62.4)
C_TP	4/50 (3.83)	4/50 (19.5)	6/50 (1.41)	22/50 (13.7)
C_pSPI	38/50 (154)	25/50 (141)	13/50 (56.1)	13/50 (122)
C_TMBS	39/50 (21.2)	25/50 (82.2)	16/50 (7.21)	41/50 (40.3)
C_RBR	10/50 (0.87)	1/50 (54.7)	7/50 (1.85)	27/50 (7.25)
C_TSA	18/50 (37.9)	14/50 (564)	49/50 (5.18)	50/50 (43.1)
C_HYB	28/50 (50.2)	18/50 (451)	49/50 (7.04)	48/50 (82.8)
MS	0/50 (0.88)	1/50 (19.9)	0/50 (0.52)	1/50 (10.6)
MBS	1/50 (0.70)	2/50 (82.9)	7/50 (1.08)	2/50 (11.3)
RBR	0/50 (0.35)	0/50 (20.4)	0/50 (0.13)	5/50 (22.9)
SA	2/50 (1.53)	4/50 (76.3)	12/50 (2.48)	7/50 (7.53)
HYB	2/50 (0.82)	4/50 (62.1)	11/50 (0.93)	8/50 (17.8)

Table 5.2 Performance comparison of the different heuristics on the 4 most difficult infinitesimally rigid non-negative factorizations introduced in section 3.2.1. The asterisk indicates the matrices that the authors of [KK21] were unable to factorize with 10 runs of the SA heuristic. The horizontal dotted line separates the heuristics of this thesis with the heuristics of [Van+16].

5.2.3 Mid-term review

We can conclude from both Table 5.1 and Table 5.2 that the conic-based heuristics compete favorably with the algorithms proposed in [Van+16] and appears to be more robust in some difficult settings. The multi-start heuristics (`C_TMBS`, `C_RBR`, `C_TSA`, `C_HYB`) clearly stand out for their better exploration trade-off: they achieve better performances in a reduced time. For the following numerical experiment, we decided to keep only the most promising multi-start heuristics (both in the BCD and conic context) : `C_TMBS`, `C_RBR`, `C_TSA`, `C_HYB`, `RBR`, `SA`, and `HYB`.

One could think that we have over-advantaged the conic-based heuristics in some way. In reality, the opposite is true. For all the exploration paradigms, the BCD version can always enjoy more (full or partial) re-initializations than their conic counterparts: multi-best-start (`C_TMBS` : 11 VS 200 : `MBS`), rank-by-rank (`C_RBR` : 5r VS 10r : `RBR`) and simulated annealing (`C_TSA` : 50 VS 1000 : `SA`). This observation confirms the good ‘catalyzer’ nature of the full-space conic approach: each initialization tends to be better exploited towards exactNMFs in the conic context. This is particularly promising when considering problems where BCD-based heuristics are struggling to find exactNMFs.

5.3 Main numerical experiments

In this section, we test the conic-based heuristics on particular settings where BCD-based heuristics are having serious trouble finding exactNMFs.

5.3.1 Generic n -gons

Let us take a look at the case of generic n -gons introduced in section 3.2.2. In particular, we want here to put conjecture (3.12) to the test. Before trying to disprove it empirically, let us first figure out how good the heuristics are in finding the conjectured factorizations.

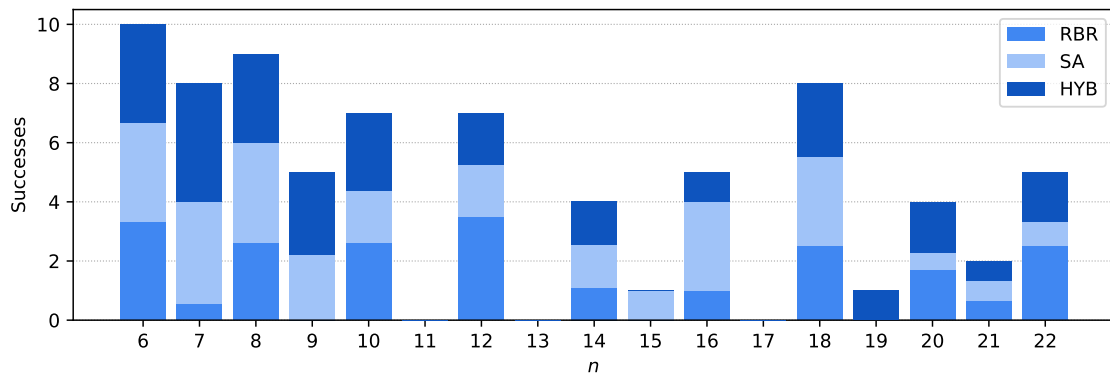


Figure 5.6 Number of generic n -gons for which at least one exactNMF corresponding to conjecture (3.12) was found with at least one BCD-based heuristics. Colors indicate the proportion of at least one success for the different heuristics. Note that each heuristic was launched 20 times for each matrix.

In Figure 5.6, we see that as dimensions increase, BCD-based heuristics are finding it increasingly difficult to reach exactNMFs corresponding to the conjecture. Especially for most of the odd dimensions, few or sometimes none are found.

We, therefore, decided to put the conic- and BCD-based heuristics in competition with each other on the following odd dimensions: 7, 9, 11, 13, 15. As can be observed in Figure 5.7, the conic-based heuristics seem to perform better on these difficult random instances. For small dimensions, both approaches are successful and, as dimensions increase, successful factorizations are mainly found by a conic-based heuristic.

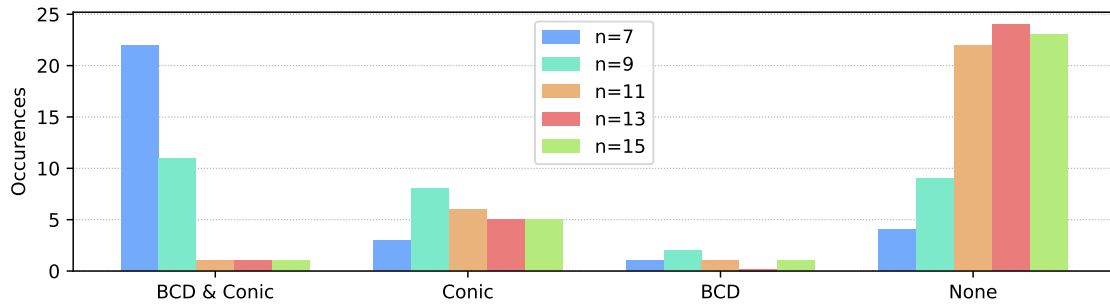


Figure 5.7 Family analysis (BCD VS Conic) of the results on 30 generic n -gons for each dimension in $[7, 9, 11, 13, 15]$ with factorization rank corresponding to conjecture (3.12). Each heuristic was launched 20 times for each matrix. If at least one success is observed for a given matrix, we make a distinction here between 3 categories: *BCD & Conic* (at least one success for both the conic- and BCD-based heuristics), *Conic* (success(es) arise only on the conic side) and *BCD* (success(es) arise only on the BCD side).

Table 5.3 shows an aggregated and quantitative view of the same results. It is here clear that when a success is observed, in almost 93% of the cases, at least one conic-based heuristic was able to find the desired factorization. But most importantly, for 18% of all the matrices, only the conic family was able to find the desired factorization increasing the total success rate from 27% to 45%.

		CONIC		
		I	0	total
BCD	I	36	5	41
	0	27	82	109
total		63	87	150

Table 5.3 A 'confusion matrix'-like representation of the results of the numerical experiment described in Figure 5.7.

Note that, in addition, for matrices for which no factorization was found, in 80% of the cases, a conic-based heuristic found a better factorization (in the sense of the residual norm) than all the BCD-based ones. All the previous comments tend to confirm the better robustness of the conic approach (at least for this particular setting).

As an attempt to compare the robustness of each heuristic individually, we define the performance metric m_h for heuristic h as

$$m_h = \frac{1}{N} \sum_{i=1}^N \frac{a_i^{min}}{a_i^h} \quad (5.4)$$

where a_i^h is the average least square approximation error achieved by heuristic h on the 30 runs for the i th matrix and a_i^{min} is the best average least square approximation error achieved for the i th matrix. Hence this metric is between 0 and 1, and bigger values are better. We can observe in Figure 5.8 that the **C_TSA** and **C_HYB** heuristics clearly outperforms their competitors to find exactNMFs in the generic n -gon context.

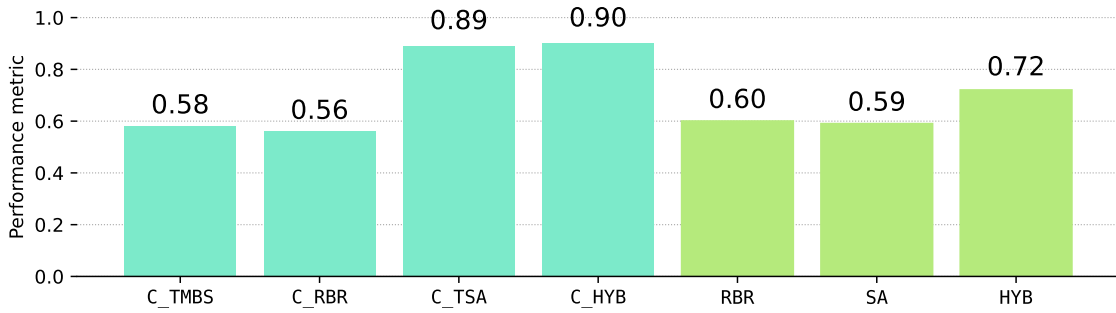


Figure 5.8 Performance metric (5.4) of each of the conic- and BCD-based heuristics involved in the numerical experiment described in Figure 5.7.

For the same dimensions and with the same number of runs, we tried to disprove conjecture (3.12) but without success. This gives even more strength to the conjecture.

5.3.2 Random linear Euclidean distance matrices

Following the same methodology as in the previous section, we address conjecture (3.15) concerning random LEDMs which are introduced in section 3.2.3.

In Figure 5.9, we see that for some specific dimensions ($n = 10, 12$ and 16), BCD-based heuristics struggle in finding exactNMFs corresponding to the conjecture. For these difficult random instances, as can be observed in Figure 5.10, the conic-based heuristics seem to perform better in the sense that they are able to find exactNMFs where BCD-based heuristics cannot. Table 5.4 reports that when a success is observed, in almost 89% of the cases, at least one conic-based heuristic was able to find the desired factorization. But most importantly, for 33% of all the matrices, only the conic family was able to find the desired factorization increasing the total success rate from 56% to 89%.

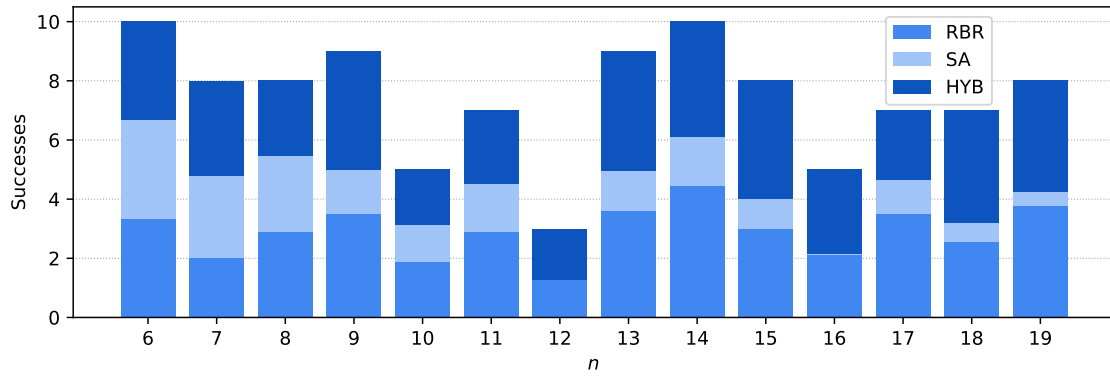


Figure 5.9 Number of random LEDMs of size $n \times n$ for which at least one exactNMF corresponding to conjecture (3.15) was found with at least one BCD-based heuristics. Colors indicate the proportion of at least one success for the different heuristics. Note that each heuristic was launched 20 times for each matrix.

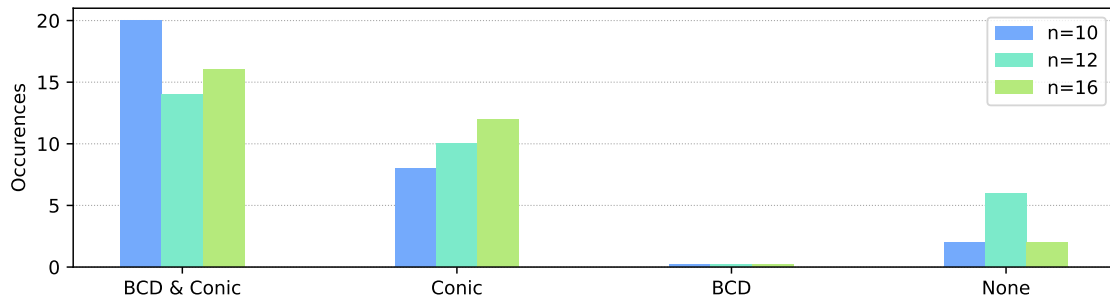


Figure 5.10 Family analysis (BCD VS Conic) of the results on 30 random LEDMs for each dimension in $[10, 12, 16]$ with factorization rank corresponding to conjecture (3.15). Each heuristic was launched 20 times for each matrix. If at least one success is observed for a given matrix, we make a distinction here between 3 categories: *BCD & Conic* (at least one success for both the conic- and BCD-based heuristics), *Conic* (success(es) arise only on the conic side) and *BCD* (success(es) arise only on the BCD side).

		CONIC		
		I	0	total
BCD	I	50	0	50
	0	30	10	40
total		80	10	90

Table 5.4 A 'confusion matrix'-like representation of the results of the numerical experiment described in Figure 5.10.

Regarding the average relative performance of the heuristics in this particular setting, it is clear from Figure 5.11 that the conic-based heuristics outperform their BCD-based counterparts. It is interesting to note that the performances of some heuristics vary significantly between the random LEDM setting and the generic n -gon setting, especially for C_RBR , C_TMBS , and SA . But, in both cases, conic-based heuristics wins.

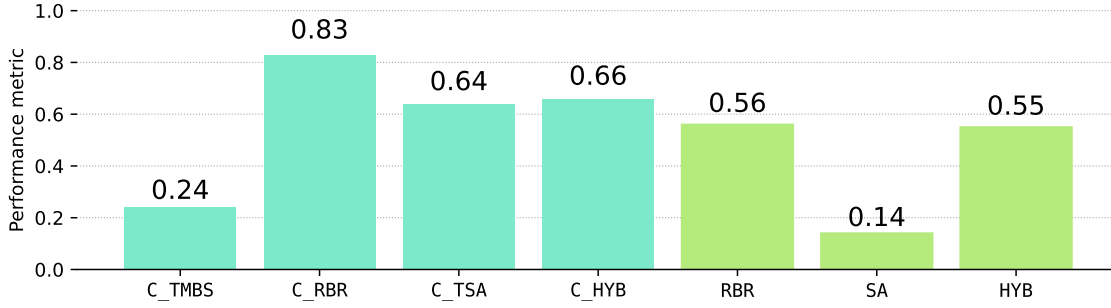


Figure 5.11 Performance metric (5.4) of each of the conic- and BCD-based heuristics involved in the numerical experiment described in Figure 5.10.

Note that, again, for matrices for which no factorization was found, in 70% of the cases, a conic-based heuristic found a better factorization (in the sense of the residual norm) than all the BCD-based ones.

Due to lack of time, we are leaving conjecture (3.12) to be put to the test in future research.

5.3.3 Open question on a UDISJ matrix

As presented in section 3.2.4, for the UDISJ matrix U_3 , it is unclear whether $\text{rank}_+(U_3) = 7$ or $\text{rank}_+(U_3) = 8$. As the conjecture in [Van+16] suggests, it is strongly believed that $\text{rank}_+(U_3) = 8$. In fact, the BCD-based heuristics cannot compute any factorization with a small error: for $r = 7$, the best solution found has relative error $\|U_3 - WH\|_F / \|U_3\|_F = 0.0836$.

When running the conic-based heuristics on U_3 with the same factorization rank $r = 7$, we observe that all the runs are (approximately) distributed among 3 local minima: 0.1059 (7%), 0.0867 (41%), and **0.0706** (52%). The smallest relative error is thereby a little bit improved but is still too big to hope for an exactNMF of rank $r = 7$. On the contrary, these results strengthen even more the conjecture in [Van+16].

Discussion and Conclusion

In this thesis, we have introduced heuristics combining the conic full-space approach of [Lep+21] and the block-coordinate approach of [Van+16] to address the *ExactNMF* problem. Using well-chosen numerical experiments, we have shown that they constitute (at least) a successful complement to the state-of-the-art heuristics and local search algorithms that have already been proposed in the NMF context.

Towards conic catalyzers. Based on the observation that the conic Frank-Wolfe (CFW) algorithm proposed to solve the conic NMF reformulations struggles to provide high-quality exactNMFs without sparsity jumps (SPI) but has a high and interesting degree of mobility in its first iterations, it has been given a facilitator mission towards exactNMFs. On this basis, the conic-based heuristics introduced in Chapter 4 operate in two stages. They start with a conic catalyzer and then further refine the solution by means of fast state-of-the-art BCD algorithms. To link both approaches in an optimal way, i.e. reducing as much as possible the computational load coming from the numerous calls to IPMs in the conic part while preserving fruitful full-space conic exploration, we designed smart *plateau* and *threshold* switches. These ‘convergence-aware’ switches have been integrated into several exploration strategies: multi-best-start, rank-by-rank, and simulated annealing. The role of the full-space conic approach has thereby been limited to the identification of a promising region of the NMF solution space leaving systematically the end of the job to a BCD algorithm.

Convincing numerical experiments. The aim of this thesis was to compute exact-NMFs where the existing heuristics cannot (whatever the computational load). In Chapter 5, we found out that in both the generic n -gon and the random LEDM settings, BCD-based methods have proved incapable of factoring many matrices. In many of these situations, the conic-based heuristics have reached better factorizations and even found exactNMFs, i.e. factorizations with a relative approximation error smaller than 10^{-6} . The performance of the different methods seems to depend

on the class of matrices considered. In a vast majority of the cases, at least one conic-based heuristic outperforms all its BCD counterparts.

Perspectives. The favorable results of Chapter 5 were obtained with conic-based heuristics involving fewer full or partial re-initializations along the optimization process than their BCD counterparts. This indicates that conic full-space exploration tends to be more robust in the sense that each starting point is better exploited on average by a conic full-space exploration. BCD-based heuristics rely more on their number of starting points to reach good performances in difficult settings. Since the quality and the number of initializations remain key in the conic context, implementing conic-based heuristics with more full or partial re-initialization could lead to even better results. Unfortunately, although some small improvements have been achieved with respect to [Lep+21], the poor scalability of the iterations of the CFW algorithm remains a considerable limitation of the conic-based approach. As we have already mentioned, in this thesis, the conic optimization sub-problems (LMO) of the CFW algorithm were solved using the IPMs of the Mosek optimization software as a ‘black box’. We believe that the best direction of research towards better scalability would be to design a more adaptative and sparing optimization method to address the LMOs. For example, should all the conic sub-problems be solved up to optimality? Can some type of suboptimality lead to a better time/precision trade-off while preserving good convergence properties? With improved scalability, we could reach higher dimensional problems. In that regard, as BCD-based heuristics struggle to find exactNMFs in higher dimensional settings [Van+16], it could be interesting to observe whether the better robustness of the conic full-space exploration is confirmed for higher dimensional problems.

References

- [AG19] Andersen Man Shun Ang and Nicolas Gillis. “Accelerating nonnegative matrix factorization algorithms using extrapolation”. In: *Neural computation* 31.2 (2019), pp. 417–439 (Cited on page 20).
- [ApS19] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 9.0*. 2019. URL: <http://docs.mosek.com/9.0/toolbox/index.html> (Cited on page 31).
- [Aro+12] Sanjeev Arora, Rong Ge, Ravindran Kannan, and Ankur Moitra. “Computing a nonnegative matrix factorization—provably”. In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. 2012, pp. 145–162 (Cited on pages v, 6).
- [AQG19] Syed Muhammad Atif, Sameer Qazi, and Nicolas Gillis. “Improved SVD-based initialization for nonnegative matrix factorization using low-rank correction”. In: *Pattern Recognition Letters* 122 (2019), pp. 53–59 (Cited on page 56).
- [Bar+04] Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. “An information statistics approach to data stream and communication complexity”. In: *Journal of Computer and System Sciences* 68.4 (2004), pp. 702–732 (Cited on page 42).
- [BL09] LeRoy B Beasley and Thomas J Laffey. “Real rank versus nonnegative rank”. In: *Linear Algebra and its Applications* 431.12 (2009), pp. 2330–2335 (Cited on page 41).
- [BN01] Aharon Ben-Tal and Arkadi Nemirovski. “On polyhedral approximations of the second-order cone”. In: *Mathematics of Operations Research* 26.2 (2001), pp. 193–205 (Cited on page 39).
- [BP94] Abraham Berman and Robert J Plemmons. *Nonnegative matrices in the mathematical sciences*. SIAM, 1994 (Cited on page 10).
- [Ber+07] Michael W Berry, Murray Browne, Amy N Langville, V Paul Pauca, and Robert J Plemmons. “Algorithms and applications for approximate nonnegative matrix factorization”. In: *Computational statistics & data analysis* 52.1 (2007), pp. 155–173 (Cited on page 15).
- [BP16] Gábor Braun and Sebastian Pokutta. “Common information and unique disjointness”. In: *Algorithmica* 76 (2016), pp. 597–629 (Cited on page 42).
- [CR10] Enrico Carlini and Fabio Rapallo. “Probability matrices, non-negative rank, and parameterization of mixture models”. In: *Linear Algebra and its Applications* 433.2 (2010), pp. 424–432 (Cited on pages v, 35).
- [CP10] Arkadev Chattopadhyay and Toniann Pitassi. “The story of set disjointness”. In: *ACM SIGACT News* 41.3 (2010), pp. 59–85 (Cited on page 41).
- [CLS21] Zhaoyue Chen, Mokhwa Lee, and Yifan Sun. “Continuous Time Frank-Wolfe Does Not Zig-Zag, But Multistep Methods Do Not Accelerate”. In: *arXiv preprint arXiv:2106.05753* (2021) (Cited on page 59).
- [CS22] Zhaoyue Chen and Yifan Sun. “Accelerating Frank-Wolfe via Averaging Step Directions”. In: *arXiv preprint arXiv:2205.11794* (2022) (Cited on page 59).

- [Chi+17] Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. “On rationality of nonnegative matrix factorization”. In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 1290–1305 (Cited on page 6).
- [CL08] Moody T Chu and Matthew M Lin. “Low-dimensional polytope approximation and its applications to nonnegative matrix factorization”. In: *SIAM Journal on Scientific Computing* 30.3 (2008), pp. 1131–1155 (Cited on page 12).
- [CP09] Andrzej Cichocki and Anh-Huy Phan. “Fast local algorithms for large scale nonnegative matrix and tensor factorizations”. In: *IEICE transactions on fundamentals of electronics, communications and computer sciences* 92.3 (2009), pp. 708–721 (Cited on page 18).
- [CZA07] Andrzej Cichocki, Rafal Zdunek, and Shun-ichi Amari. “Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization”. In: *Independent Component Analysis and Signal Separation: 7th International Conference, ICA 2007, London, UK, September 9-12, 2007. Proceedings 7*. Springer. 2007, pp. 169–176 (Cited on pages 18, 19).
- [CR93] Joel E Cohen and Uriel G Rothblum. “Nonnegative ranks, decompositions, and factorizations of nonnegative matrices”. In: *Linear Algebra and its Applications* 190 (1993), pp. 149–168 (Cited on pages 5, 12).
- [CCZ10] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. “Extended formulations in combinatorial optimization”. In: *4OR* 8.1 (2010), pp. 1–48 (Cited on page 38).
- [Coo00] Stephen Cook. “The P versus NP problem”. In: *Clay Mathematics Institute* 2 (2000) (Cited on page 6).
- [DeM72] Ralph DeMarr. “Nonnegative matrices with nonnegative inverses”. In: *Proceedings of the American Mathematical Society* 35.1 (1972), pp. 307–308 (Cited on page 4).
- [Dew22] Julien Dewez. “Computational approaches for lower bounds on the nonnegative rank”. PhD thesis. Ph. D. diss., UCLouvain, 2022 (Cited on pages 35, 37).
- [DGG21] Julien Dewez, Nicolas Gillis, and François Glineur. “A geometric lower bound on the extension complexity of polytopes based on the f-vector”. In: *Discrete Applied Mathematics* 303 (2021), pp. 22–38 (Cited on page 7).
- [DV16] Xuan Vinh Doan and Stephen Vavasis. “Finding the largest low-rank clusters with Ky Fan 2-k-norm and ℓ_1 -norm”. In: *SIAM Journal on Optimization* 26.1 (2016), pp. 274–312 (Cited on page 9).
- [DS03] David Donoho and Victoria Stodden. “When does non-negative matrix factorization give a correct decomposition into parts?” In: *Advances in neural information processing systems* 16 (2003) (Cited on pages 4, 10).
- [Esp21] Flavia Esposito. “A review on initialization methods for nonnegative matrix factorization: towards omics data experiments”. In: *Mathematics* 9.9 (2021), p. 1006 (Cited on page 21).
- [FP15] Hamza Fawzi and Pablo A Parrilo. “Lower bounds on nonnegative rank via nonnegative nuclear norms”. In: *Mathematical Programming* 153 (2015), pp. 41–66 (Cited on pages 7, 9).
- [FP16] Hamza Fawzi and Pablo A Parrilo. “Self-scaled bounds for atomic cone ranks: applications to nonnegative rank and cp-rank”. In: *Mathematical Programming* 158 (2016), pp. 417–465 (Cited on page 7).
- [Fio+13] Samuel Fiorini, Volker Kaibel, Kanstantsin Pashkovich, and Dirk Oliver Theis. “Combinatorial bounds on nonnegative rank and extended formulations”. In: *Discrete mathematics* 313.1 (2013), pp. 67–83 (Cited on pages v, 7, 36, 37).

- [Fio+12] Samuel Fiorini, Serge Massar, Sebastian Pokutta, Hans Raj Tiwary, and Ronald De Wolf. “Linear vs. semidefinite extended formulations: exponential separation and strong lower bounds”. In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. 2012, pp. 95–106 (Cited on page 42).
- [FRT12] Samuel Fiorini, Thomas Rothvoß, and Hans Raj Tiwary. “Extended formulations for polygons”. In: *Discrete & computational geometry* 48.3 (2012), pp. 658–668 (Cited on pages 39, 40).
- [FW56] Marguerite Frank and Philip Wolfe. “An algorithm for quadratic programming”. In: *Naval research logistics quarterly* 3.1-2 (1956), pp. 95–110 (Cited on page 30).
- [Fu+19] Xiao Fu, Kejun Huang, Nicholas D 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 (Cited on page 4).
- [Gil12] Nicolas Gillis. “Sparse and unique nonnegative matrix factorization through data preprocessing”. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 3349–3386 (Cited on page 5).
- [Gil14] Nicolas Gillis. “The why and how of nonnegative matrix factorization”. In: *Connections* 12.2 (2014) (Cited on pages 16, 21).
- [Gil17] Nicolas Gillis. “Introduction to nonnegative matrix factorization”. In: *arXiv preprint arXiv:1703.00663* (2017) (Cited on pages 9, 11, 60).
- [Gil20] Nicolas Gillis. *Nonnegative matrix factorization*. SIAM, 2020 (Cited on pages 2, 5, 8, 9, 11, 18, 20, 21, 42).
- [GG08a] Nicolas Gillis and François Glineur. “Nonnegative factorization and the maximum edge biclique problem”. In: *arXiv preprint arXiv:0810.4225* (2008) (Cited on pages 14, 15, 16, 19, 20).
- [GG08b] Nicolas Gillis and François Glineur. “Nonnegative Matrix Factorization and Underapproximation”. In: *Communication at 9th International Symposium on Iterative Methods in Scientific Computing*. Lille, France, 2008 (Cited on page 18).
- [GG10] Nicolas Gillis and François Glineur. “Using underapproximations for sparse nonnegative matrix factorization”. In: *Pattern recognition* 43.4 (2010), pp. 1676–1687 (Cited on page 3).
- [GG12a] Nicolas Gillis and François Glineur. “Accelerated multiplicative updates and hierarchical ALS algorithms for nonnegative matrix factorization”. In: *Neural computation* 24.4 (2012), pp. 1085–1105 (Cited on page 20).
- [GG12b] Nicolas Gillis and François Glineur. “On the geometric interpretation of the nonnegative rank”. In: *Linear Algebra and its Applications* 437.11 (2012), pp. 2685–2712 (Cited on pages v, 7, 24, 36, 39, 41).
- [GKP14] Nicolas Gillis, Da Kuang, and Haesun Park. “Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization”. In: *IEEE Transactions on Geoscience and Remote Sensing* 53.4 (2014), pp. 2066–2078 (Cited on page 56).
- [GV13a] Nicolas Gillis and Stephen A 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 (Cited on page 56).
- [Goe15] Michel X Goemans. “Smallest compact formulation for the permutahedron”. In: *Mathematical Programming* 153.1 (2015), pp. 5–11 (Cited on page 39).
- [GV13b] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013 (Cited on page 10).

- [GS00] Luigi Grippo and Marco Sciandrone. “On the convergence of the block nonlinear Gauss–Seidel method under convex constraints”. In: *Operations research letters* 26.3 (2000), pp. 127–136 (Cited on page 17).
- [HS20] Aritra Narayan Hisabia and Manideepa Saha. “On Properties of Semipositive Cones and Simplicial Cones”. In: *The Electronic Journal of Linear Algebra* 36 (2020), pp. 764–772 (Cited on page 11).
- [HVB11] Ngoc-Diep Ho, Paul Van Dooren, and Vincent D Blondel. “Descent methods for nonnegative matrix factorization”. In: *Numerical Linear Algebra in Signals, Systems and Control*. Springer, 2011, pp. 251–293 (Cited on pages 14, 18).
- [Hru12] Pavel Hrubeš. “On the nonnegative rank of distance matrices”. In: *Information Processing Letters* 112.11 (2012), pp. 457–461 (Cited on page 41).
- [JT11a] Andreas Janecek and Ying Tan. “Using population based algorithms for initializing nonnegative matrix factorization”. In: *Advances in Swarm Intelligence: Second International Conference, ICSI 2011, Chongqing, China, June 12-15, 2011, Proceedings, Part II* 2. Springer. 2011, pp. 307–316 (Cited on page 22).
- [JT11b] Andreas Janecek and Ying Tan. “Iterative improvement of the multiplicative update nmf algorithm using nature-inspired optimization”. In: *2011 Seventh International Conference on Natural Computation*. Vol. 3. IEEE. 2011, pp. 1668–1672 (Cited on page 22).
- [KP11a] Volker Kaibel and Kanstantsin Pashkovich. “Constructing extended formulations from reflection relations”. In: *Integer Programming and Combinatorial Optimization: 15th International Conference, IPCO 2011, New York, NY, USA, June 15-17, 2011. Proceedings* 15. Springer. 2011, pp. 287–300 (Cited on page 39).
- [KP13] Volker Kaibel and Kanstantsin Pashkovich. “Constructing extended formulations from reflection relations”. In: *Facets of Combinatorial Optimization: Festschrift for Martin Grötschel* (2013), pp. 77–100 (Cited on page 39).
- [KS92] Bala Kalyanasundaram and Georg Schintger. “The probabilistic communication complexity of set intersection”. In: *SIAM Journal on Discrete Mathematics* 5.4 (1992), pp. 545–557 (Cited on page 42).
- [KP08] Hyunsoo Kim and Haesun Park. “Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method”. In: *SIAM journal on matrix analysis and applications* 30.2 (2008), pp. 713–730 (Cited on page 17).
- [KP11b] Jingu Kim and Haesun Park. “Fast nonnegative matrix factorization: An active-set-like method and comparisons”. In: *SIAM Journal on Scientific Computing* 33.6 (2011), pp. 3261–3281 (Cited on page 17).
- [KW12] Nathan Krislock and Henry Wolkowicz. *Euclidean distance matrices and applications*. Springer, 2012 (Cited on page 41).
- [KK21] Robert Krone and Kaie Kubjas. “Uniqueness of nonnegative matrix factorizations by rigidity theory”. In: *SIAM Journal on Matrix Analysis and Applications* 42.1 (2021), pp. 134–164 (Cited on pages 38, 61, 62).
- [KSZ22] Matthew Kwan, Lisa Sauermann, and Yufei Zhao. “Extension complexity of low-dimensional polytopes”. In: *Transactions of the American Mathematical Society* 375.6 (2022), pp. 4209–4250 (Cited on pages 38, 40).
- [Lac16] Simon Lacoste-Julien. “Convergence rate of frank-wolfe for non-convex objectives”. In: *arXiv preprint arXiv:1607.00345* (2016) (Cited on page 31).
- [Lau+08] Hans Laurberg et al. “Theorems on positive data: On the uniqueness of NMF”. In: *Computational intelligence and neuroscience* (2008) (Cited on page 4).

- [LH95] Charles L Lawson and Richard J Hanson. *Solving least squares problems*. SIAM, 1995 (Cited on page 17).
- [LS96] Daniel Lee and H Sebastian Seung. “Unsupervised learning by convex and conic coding”. In: *Advances in neural information processing systems* 9 (1996) (Cited on page 10).
- [LS99] Daniel Lee and H Sebastian Seung. “Learning the parts of objects by non-negative matrix factorization”. In: *Nature* 401.6755 (1999), pp. 788–791 (Cited on pages 1, 3, 4, 10, 15).
- [LS00] Daniel Lee and H Sebastian Seung. “Algorithms for non-negative matrix factorization”. In: *Advances in neural information processing systems* 13 (2000) (Cited on page 15).
- [Lep+21] Valentin Leplat, Yurii Nesterov, Nicolas Gillis, and François Glineur. “Exact Nonnegative Matrix Factorization via Conic Optimization”. In: *arXiv preprint arXiv:2105.13646* (2021) (Cited on pages v, 28, 29, 30, 31, 32, 43, 46, 56, 57, 58, 60, 61, 69, 70).
- [Li+20] Bingcong Li, Mario Coutino, Georgios B Giannakis, and Geert Leus. “How does momentum help Frank Wolfe?” In: *arXiv preprint arXiv:2006.11116* (2020) (Cited on page 59).
- [Li+21] Bingcong Li, Mario Coutino, Georgios B Giannakis, and Geert Leus. “A momentum-guided frank-wolfe algorithm”. In: *IEEE Transactions on Signal Processing* 69 (2021), pp. 3597–3611 (Cited on page 59).
- [Lin07] Chih-Jen Lin. “On the convergence of multiplicative update algorithms for nonnegative matrix factorization”. In: *IEEE Transactions on Neural Networks* 18.6 (2007), pp. 1589–1596 (Cited on page 16).
- [LC10] Matthew M Lin and Moody T Chu. “On the nonnegative rank of Euclidean distance matrices”. In: *Linear algebra and its applications* 433.3 (2010), pp. 681–689 (Cited on page 41).
- [Lov89] László Lovász. *Communication complexity: A survey*. Forschungsinst. für Diskrete Mathematik. Inst. für Ökonometrie u. Operations ..., 1989 (Cited on page v).
- [Mac67] J MacQueen. “Classification and analysis of multivariate observations”. In: *5th Berkeley Symp. Math. Statist. Probability*. University of California Los Angeles LA USA. 1967, pp. 281–297 (Cited on page 13).
- [Moi16] Ankur Moitra. “An almost optimal algorithm for computing nonnegative rank”. In: *SIAM Journal on Computing* 45.1 (2016), pp. 156–173 (Cited on page 6).
- [MSV03] David Mond, Jim Smith, and Duco Van Straten. “Stochastic factorizations, sandwiched simplices and the topology of the space of explanations”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 459.2039 (2003), pp. 2821–2845 (Cited on page 61).
- [Ngo08] Ho Ngoc-Diep. “Nonnegative matrix factorization algorithms and applications”. PhD thesis. Ph. D. Dissertation. PhD thesis, Université catholique de Louvain., 2008 (Cited on pages 1, 13).
- [Nis91] Noam Nisan. “Lower bounds for non-commutative computation”. In: *Proceedings of the twenty-third annual ACM symposium on Theory of computing*. 1991, pp. 410–418 (Cited on page v).
- [OVW14] Michael Oelze, Arnaud Vandaele, and Stefan Weltge. “Computing the extension complexities of all 4-dimensional 0/1-polytopes”. In: *arXiv preprint arXiv:1406.4895* (2014) (Cited on pages 7, 39).
- [PT94] Pentti Paatero and Unto Tapper. “Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values”. In: *Environmetrics* 5.2 (1994), pp. 111–126 (Cited on pages 1, 16).
- [PP14] Arnau Padrol and Julian Pfeifle. “Polygons as slices of higherdimensional polytopes”. In: *arXiv preprint arXiv:1404.2443* (2014) (Cited on page 40).

- [PA71] Azaria Paz and Werner Rheinboldt (Auth.) *Introduction to Probabilistic Automata*. Computer Science and Applied Mathematics. Elsevier Inc, Academic Press Inc, 1971. ISBN: 9780125476508; 0125476507. URL: libgen.li/file.php?md5=5b9fa6eedc87fe21f4c68d69e1861841 (Cited on page v).
- [Pir96] Marc Pirlot. “General local search methods”. In: *European journal of operational research* 92.3 (1996), pp. 493–511 (Cited on page 26).
- [Raz92] A. Razborov. “On the distributional complexity”. In: *Theoretical Computer Science* 106 (1992), pp. 385–390 (Cited on page 42).
- [Rot17] Thomas Rothvoß. “The matching polytope has exponential extension complexity”. In: *Journal of the ACM (JACM)* 64.6 (2017), pp. 1–19 (Cited on page 38).
- [Shi14a] Yaroslav Shitov. “Sublinear extensions of polygons”. In: *arXiv preprint arXiv:1412.0728* (2014) (Cited on page 40).
- [Shi14b] Yaroslav Shitov. “An upper bound for nonnegative rank”. In: *Journal of Combinatorial Theory, Series A* 122 (2014), pp. 126–132 (Cited on pages 7, 40, 41).
- [Shi15] Yaroslav Shitov. “Nonnegative rank depends on the field”. In: *arXiv preprint arXiv:1505.01893* (2015) (Cited on page 6).
- [Shi16] Yaroslav Shitov. “Nonnegative rank depends on the field II”. In: *arXiv preprint arXiv:1605.07173* (2016) (Cited on page 6).
- [SC10] Amit Singer and Mihai Cucuringu. “Uniqueness of low-rank matrix completion by rigidity theory”. In: *SIAM Journal on Matrix Analysis and Applications* 31.4 (2010), pp. 1621–1641 (Cited on page 38).
- [SGB05] Age K Smilde, Paul Geladi, and Rasmus Bro. *Multi-way analysis: applications in the chemical sciences*. John Wiley & Sons, 2005 (Cited on page 16).
- [Sta+07] Kurt Stadlthanner, Dominik Lutter, Fabian J Theis, Elmar Wolfgang Lang, Ana Maria Tomé, Petia Georgieva, and Carlos Garcia Puntonet. “Sparse nonnegative matrix factorization with genetic algorithms for microarray analysis”. In: *2007 International Joint Conference on Neural Networks*. IEEE, 2007, pp. 294–299 (Cited on page 22).
- [SS93] Patrick Suppes and Patrick Suppes. “When are probabilistic explanations possible?” In: *Models and methods in the philosophy of science: selected essays* (1993), pp. 141–148 (Cited on pages v, 35).
- [TOG17] Csaba D Toth, Joseph O’Rourke, and Jacob E Goodman. *Handbook of discrete and computational geometry*. CRC press, 2017 (Cited on page 38).
- [VGG17] Arnaud Vandaele, Nicolas Gillis, and François Glineur. “On the linear extension complexity of regular n-gons”. In: *Linear Algebra and its Applications* 521 (2017), pp. 217–239 (Cited on pages 7, 39).
- [Van+16] Arnaud Vandaele, Nicolas Gillis, François Glineur, and Daniel Tuytens. “Heuristics for exact nonnegative matrix factorization”. In: *Journal of Global Optimization* 65 (2016), pp. 369–400 (Cited on pages v, 21, 24, 26, 28, 36, 37, 39, 40, 41, 42, 49, 53, 60, 62, 63, 67, 69, 70).
- [Vav10] Stephen A Vavasis. “On the complexity of nonnegative matrix factorization”. In: *SIAM Journal on Optimization* 20.3 (2010), pp. 1364–1377 (Cited on pages v, 6, 9).
- [WW93] Wolfgang Weil and J Wieacker. *Stochastic geometry, Handbook of convex geometry, Vol. A, B, 1391–1438*. 1993 (Cited on page 40).
- [Wik22a] Wikipedia contributors. *Frank–Wolfe algorithm* — *Wikipedia, The Free Encyclopedia*. [Online; accessed 26-April-2023]. 2022. URL: https://en.wikipedia.org/w/index.php?title=Frank%E2%80%99s_algorithm&oldid=1127852423 (Cited on page 32).

- [Wik22b] Wikipedia contributors. *Local search (optimization)* — *Wikipedia, The Free Encyclopedia*. [Online; accessed 12-April-2023]. 2022. URL: [https://en.wikipedia.org/w/index.php?title=Local_search_\(optimization\)&oldid=1101424577](https://en.wikipedia.org/w/index.php?title=Local_search_(optimization)&oldid=1101424577) (Cited on page 15).
- [Yan88] Mihalis Yannakakis. “Expressing combinatorial optimization problems by linear programs”. In: *Proceedings of the twentieth annual ACM symposium on Theory of computing*. 1988, pp. 223–228 (Cited on pages v, 36, 37).
- [Yao79] Andrew Chi-Chih Yao. “Some complexity questions related to distributive computing (preliminary report)”. In: *Proceedings of the eleventh annual ACM symposium on Theory of computing*. 1979, pp. 209–213 (Cited on page 36).
- [Zhu+18] Zhihui Zhu, Qiuwei Li, Gongguo Tang, and Michael B Wakin. “Global optimality in low-rank matrix optimization”. In: *IEEE Transactions on Signal Processing* 66.13 (2018), pp. 3614–3628 (Cited on page 9).

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/epl