Matrix factorization ranks via polynomial optimization

Andries Steenkamp

Matrix factorization ranks via polynomial optimization

Andries Steenkamp *

Abstract In light of recent data science trends, new interest has fallen in alternative matrix factorizations. By this, we mean various ways of factorizing particular data matrices so that the factors have special properties and reveal insights into the original data. We are interested in the specialized ranks associated with these factorizations, but they are usually difficult to compute. In particular, we consider the nonnegative-, completely positive-, and separable ranks. We focus on a general tool for approximating factorization ranks, the moment hierarchy, a classical technique from polynomial optimization, further augmented by exploiting ideal-sparsity. Contrary to other examples of sparsity, the resulting sparse hierarchy yields equally strong, if not superior, bounds while potentially delivering a speed-up in computation.

1 Introduction and motivation for matrix factorization ranks

We live in a digital world. Data drives decisions as a never-ending stream of information engulfs our lives. An essential tool for navigating the flood of information is the ability to distill large bodies of information into actionable knowledge. A practical example is *nonnegative (NN) factorization*, applied to data represented as a matrix. A NN factorization of an entry-wise nonnegative matrix $M \in \mathbb{R}_+^{m\times n}$ is a pair of nonnegative matrices $A \in \mathbb{R}_+^{m\times r}$ and $B \in \mathbb{R}_+^{r\times n}$ for some integer $r \in \mathbb{N}$ such that:

$$M = AB. \tag{1}$$

Andries Steenkamp
CWI Amsterdam, e-mail: jajs@cwi.nl

* Centrum Wiskunde & Informatica (CWI), Amsterdam. andries.steenkamp@cwi.nl
This work is supported by the European Union's Framework Programme for Research and Innovation Horizon 2020 under the Marie Skłodowska-Curie Actions Grant Agreement No. 813211 (POEMA).

The primary object of interest is the inner dimension r of the factorization. One can always take $A = M$ and $B = I$, where I is the identity matrix, hence getting $r = n$. However, the interesting case is when $r < \frac{mn}{m+n}$. In this case, one has managed to express the $m \times n$ values of M in terms of the $(m \times r) + (r \times n)$ values of A and B, and as a result, using less storage. The smallest integer r for which this is possible is called the *nonnegative matrix factorization rank*, or just the *nonnegative rank* for short, and is mathematically defined as follows,

$$\text{rank}_+(M) := \min\{r \in \mathbb{N} : M = AB \text{ for some } A \in \mathbb{R}_+^{m\times r} \text{ and } B \in \mathbb{R}_+^{r\times n}\}. \quad (2)$$

It is not hard to see that the NN-rank is sandwiched between the classical rank and the size of the matrix, i.e.,

$$\text{rank}(M) \leq \text{rank}_+(M) \leq \min\{n, m\}.$$

However, storage space efficiency is only part of the value of NN factorization. The true power of NN factorization comes from the fact that it is an easy-to-interpret *linear dimensionality reduction* technique. To understand what we mean by this, we first re-examine the relationship between the three matrices M, A, and B in eq. (1).

Observe how the j^{th} column of M is given as a conic combination of the columns of A with weights given by the j^{th} column of B, i.e.,

$$M_{:,j} = \sum_{i=1}^{r} B_{i,j} A_{:,i}. \quad (3)$$

Because all terms involved are nonnegative, zero entries in M force the corresponding entries of the factors to be zero. Formally, for any $k \in \{1, 2, .., m\}$ and $j \in \{1, 2, .., n\}$, $M_{k,j} = 0$ if and only if $B_{i,j}A_{k,i} = 0$ for all $i \in \{1, 2, .., r\}$. Having no cancellation among factors will be useful for interpreting applications of nonnegative factorization. We will explain this further in section 1.1 with examples. Furthermore, observe that the nonnegative factorization needs not be unique. In fact, for any non-singular, nonnegative matrix $P \in \mathbb{R}_+^{r\times r}$ with nonnegative inverse P^{-1} one can produce another factorization

$$M = (AP^{-1})(PB). \quad (4)$$

An example of such a matrix P would be a permutation matrix.

Example of a nonnegative factorization

Consider the following example of a 4×4 nonnegative matrix and its nonnegative factorization from which we can deduce that $\text{rank}_+(M) = 2$, because $\text{rank}(M) = 2$:

$$M = \begin{bmatrix} 35 & 38 & 41 & 44 \\ 79 & 86 & 93 & 100 \\ 123 & 134 & 145 & 156 \\ 167 & 182 & 197 & 212 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{bmatrix} \begin{bmatrix} 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} = AB.$$

1.1 Applications of nonnegative factorization

Having introduced the nonnegative rank, we now justify its importance with three applications. What we present here is but a small fraction of the whole body of literature on nonnegative factorization. The interested reader is highly encouraged to read a recent monograph of Gillis [24] for an in-depth study of the nonnegative rank with many applications and further references. Alternatively, we invite the reader to try and conceive a few applications of their own.

Image processing

When analyzing large amounts of images, it is natural to ask if the vast bulk of images are not just combinations of a few “basic images". This raises two questions. First, how does one find or construct a set of basic images? Second, given this, hopefully small, set of basic images, how does one reproduce the original images? Lee and Seung answered both questions in [35], where they factorized a set of images of human faces into a set of typical facial features and nonnegative weights. Combining the weights and features, one approximately recovers the original faces. In this setting, the matrix M has as columns the vectorized gray-scale images of human faces, hence $M_{i,j}$ is the i^{th} pixel of the j^{th} face, with a value between 0 and 1, with 0 corresponding to black and 1 to white.

Recalling the interpretation of (3), we can think of the columns of the matrix A as (vectorized) images of human facial features, like a mouth or pair of eyes. Hence, the j^{th} image $M_{:,j}$ is a weighted sum of feature-images $A_{:,i}$ ($i \in [r]$), where the (nonnegative) weight of feature $A_{:,i}$ is given by entry $B_{i,j}$ of the matrix B.

In contradistinction to other techniques like *principle component analysis* (PCA), which possibly gives factors with negative entries, NN factorization saves us from the task of interpreting notions like “negative pixels" or “image cancellations." By “negative pixels," we mean negative factor values, i.e., $B_{i,j}A_{k,i} < 0$. This means that factor $B_{i,j}A_{:,i}$ does not just add features but also possibly erases the features added by other factors $B_{\ell,j}A_{:,\ell}$, where $\ell \neq i$. A fun by-product of these image factorizations is that one can generate new images by multiplying the matrix A with new weights different from B. However, the resulting images are not guaranteed to look like faces for a poor choice of weights.

Topic recovery and document classification

In text analysis, the matrix M is called the *word occurrence matrix*, and its entries $M_{i,j}$ are the number of times the i^{th} word occurs in the j^{th} document. This way of looking at a corpus of text is often called a "bag of words model," the sequence is ignored, and only the quantity is considered. Since word count is always nonnegative, M is nonnegative and has some NN factors A and B. The columns of matrix A take the meaning of "topics", and B gives the correct weights to recover M. Since there are no cancellations, we observe in the columns of A that certain sets of words tend to occur together, at least within the original set of documents. Moreover, we see how the documents (the columns of M) are composed from these base topics (the columns of A), with the importance of each topic given by the entries of B. One can hence use these learned topics to group or classify documents.

Linear extension complexity

This third application is different from the above two. First, we define the linear extension complexity, then we show how it relates to the nonnegative rank, and finally, we motivate its importance. The *linear extension complexity* of a polytope P is the smallest integer r for which P can be expressed as the linear image of an affine section of $\mathbb{R}^r_+$. Alternatively, the linear extension complexity can be defined as the smallest number of facets a higher dimensional polytope Q can have while still having P as a projection. In 1991 Yannakakis [43] proved that the linear extension complexity of P is equal to the nonnegative rank of the *slack matrix* associated with P. For a polytope P the slack matrix is $(d_i - c_i^T v)_{v \in \mathcal{V}, i \in \mathcal{I}}$, where $c_i \in \mathbb{R}^m,\ d_i \in \mathbb{R}$ come from the hyperplane representation of $P = \{x \in \mathbb{R}^m : c_i^T x \leq d_i \ \ (i \in \mathcal{I})\}$, and the vectors $v \in \mathbb{R}^m$ come from the extremal point representation of $P = \text{conv}(V)$. This link between nonnegative rank and linear extension complexity was instrumental in showing why many combinatorial problems, like the traveling salesman problem, could not be efficiently solved simply by lifting the associated problem polytope to higher dimensions in some clever way, see [22]. On the topic of lifting convex sets we refer the reader to the survey [20].

1.2 Commonly used notation

We group here some notation used throughout the chapter. For any positive integer $m \in \mathbb{N}$ we denote by $[m] := \{1, 2, ..., m\}$ the set consisting of the first m positive integers. For vectors $u, v \in \mathbb{R}^m$ we denote by $\langle u, v \rangle := \sum_{i \in [m]} u_i v_i$ the vector inner-product of u and v. Similarly, for matrices $A, B \in \mathbb{R}^{m \times n}$ we use the same notation to denote the Frobenius inner product $\langle A, B \rangle := \sum_{i \in [m],\ j \in [n]} A_{i,j} B_{i,j}$ of A and B. We say that a square matrix $A \in \mathbb{R}^{m \times m}$ is *positive semi-definite* (PSD), denoted by $A \succeq 0$, if and only if $v^T A v \geq 0$ for any choice of $v \in \mathbb{R}^m$. The set of all PSD matrices

of size $r \in \mathbb{N}$ is denoted by $\mathcal{S}_+^r := \{A \in \mathbb{R}^{r\times r} : A \succeq 0\}$. Analogous to real matrices that are PSD, there are complex square matrices that are *Hermitian PSD*. A complex matrix $A \in \mathbb{C}^{m\times m}$ is Hermitian PSD if and only if $v^*Av \geq 0$ for all $v \in \mathbb{C}^m$, where v^* is the complex conjugate of v. For positive integer m, we denote by $\mathcal{H}^m$ the set of all $m \times m$ Hermitian matrices. By

$$\Sigma[x] := \{\sum_{i\in[k]} \sigma_i : k \in \mathbb{N},\ \sigma_i = p_i^2 \text{ for some polynomial } p_i \in \mathbb{R}[x]\}$$

we denote the set of all sums of squares of polynomials. If the variables $x = (x_1, x_2, ..., x_m)$ are clear from the context we simply write Σ.

1.3 On computing the nonnegative rank

Given the utility of computing nonnegative factorizations, it is natural to ask the following. Is it difficult to compute the nonnegative rank for a given data matrix $M \geq 0$? This was answered in the affirmative in 2009 by Vavasis [42]. Despite being NP-hard to solve, good approximations are sometimes quite accessible. In section 2, we show a general technique for approximating the matrix factorization rank from below using tools from polynomial optimization. An alternative geometrically motivated approach is to look for a minimal *rectangle cover* for the support of M, see [25].Given a matrix $M \in \mathbb{R}^{n\times m}$, one seeks the smallest set of *rectangles*, sets of the form $R := \{\{i, j\} : i \in I \subset [n], j \in J \subset [n]\}$, such that for each nonzero entry $M_{i,j} \neq 0$, $\{i, j\}$ belongs to at least one of these rectangle.

Finding the factorization rank does not necessarily give a factorization. The method we propose in section 2 does not generally give a factorization, except in a particular case, which we consider in section 2.3, where it is possible to recover the factors. For NN factorization, several algorithms exist that iteratively compute A and B given a guessed value r. However, these algorithms only give approximate factorizations, that is, $M \approx AB$, with respect to some norm. A sufficiently good approximate NN factorization also implies an upper bound on the NN rank.

For practical problems, an approximation is often sufficient. For a detailed account of NN factorization, we refer the reader again to the book of Gillis [24].

1.4 Other factorization ranks

Above, we looked at NN factorization and some of its applications in data analysis and optimization theory. However, there are many more matrix factorization ranks, each having its intricacies, applications, and interpretations. We list a few more examples of factorization ranks to link them to NN factorization and later state some results on some of them.

Completely positive factorization

This factorization is very similar to nonnegative factorization apart from the modification that $B = A^T$. Formally, an entry-wise nonnegative matrix $M \in \mathcal{S}^m$ is *completely positive* (CP) if there exists a nonnegative matrix $A \in \mathbb{R}_+^{m \times r}$, for some integer $r \in \mathbb{N}$, such that:

$$M = AA^T. \tag{5}$$

Clearly, CP-matrices are *doubly nonnegative*, i.e., entry-wise nonnegative and *positive semi-definite* (PSD). However, these are not sufficient criteria for M to be CP, unless $m \leq 4$, see [2]. Consider the following (Example 2.9 from [2]) to see that this does not hold for $m \geq 5$.

Example of a doubly nonnegative matrix that is not CP [2]

$$M = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 \\ 1 & 0 & 0 & 1 & 6 \end{bmatrix}.$$

The nonnegativity is clear, the PSDness is checked via computing all the minors (or using a computer to check the eigenvalues). To see why the matrix is not CP, we refer to the explanation given in Berman and Shaked-Monderer's monograph [2].

Just deciding if a given matrix is CP is already an NP-hard problem; see [15]. Because the *completely positive factors* A and A^T are the same, up to transposition, the CP factorization is often called a *symmetric factorization*. We will see another example shortly at the end of this section. Similar to nonnegative rank, there is a *completely positive rank* mathematically defined as the smallest inner dimension $r \in \mathbb{N}$ for which a CP factorization of M exists, i.e.

$$\operatorname{rank}_{\text{cp}}(M) := \min\{r \in \mathbb{N} : M = AA^T \text{ for some } A \in \mathbb{R}_+^{m \times r}\}. \tag{6}$$

Clearly a matrix M is CP if and only if $\operatorname{rank}_{\text{cp}}(M) < \infty$. Hence computing the CP-rank can't be any easier than deciding if M is CP. That being said, the complexity status of computing $\operatorname{rank}_{\text{cp}}(M)$ for a CP matrix M is unknown to the best of our knowledge. Some upper bounds are known for the CP-rank: first, $\operatorname{rank}_{\text{cp}}(M) \leq m$, when $m \leq 4$, and second, $\operatorname{rank}_{\text{cp}}(M) \leq \binom{m+1}{2} - 4$ if $m \geq 5$, see [39]. In 1994 it was conjectured by Drew, Johnson, and Loewy [17] that $\operatorname{rank}_{\text{cp}}(M) \leq \lfloor \frac{m^2}{4} \rfloor$, the bound being only attained for CP matrices M that have complete bipartite support graphs. This conjecture was disproved by Bomze et al. [4, 5] two decades later, using several specially constructed counter-examples. We show in eq. (7) an example, namely $\widetilde{M}_7$ from [4], of size $m = 7$, with $\operatorname{rank}_{\text{cp}}(\widetilde{M}_7) = 14 > \lfloor \frac{49}{4} \rfloor = 12$:

$$\widetilde{M}_7 = \begin{bmatrix} 163 & 108 & 27 & 4 & 4 & 27 & 108 \\ 108 & 163 & 108 & 27 & 4 & 4 & 27 \\ 27 & 108 & 163 & 108 & 27 & 4 & 4 \\ 4 & 27 & 108 & 163 & 108 & 27 & 4 \\ 4 & 4 & 27 & 108 & 163 & 108 & 27 \\ 27 & 4 & 4 & 27 & 108 & 163 & 108 \\ 108 & 27 & 4 & 4 & 27 & 108 & 163 \end{bmatrix}. \tag{7}$$

On the more applied side, CP matrices occur in the theory of *block designs*. We omit many details here, but essentially, block designs deal with arranging distinct objects into blocks in such a way that the objects occur with certain regularity within and among the blocks. There is a direct application of block design in designing experiments where researchers wish to prevent the differences between test subjects from obfuscating the differences in outcome due to treatment, see [29] for block designs in depth and see [38] for the link between block designs and CP matrices.

From another perspective of applications, completely positive matrices are of great interest in optimization.

In 2009, Burer [7] showed that any nonconvex quadratic program with binary and continuous variables could be reformulated as a linear program over the cone of completely positive matrices. This effectively meant that many NP-hard problems could now be viewed as linear programs with CP-membership constraints. This reformulation does not make the problems any easier to solve as the difficulty is now pushed into characterizing complete positivity. However, it does allow us to attack a large class of problems by understanding a unifying thread.

For a thorough account of completely positive and copositive matrices, we refer the inquisitive reader to the monograph by Berman and Shaked-Monderer [38].

Separable rank

In the setting of quantum information theory, the state of some physical system is often characterized by a Hermitian PSD matrix $M \in \mathcal{H}^m \otimes \mathcal{H}^m$. These states are said to be *separable* if there exist vectors $a_1, ..., a_r, b_1, ..., b_r \in \mathbb{C}^m$ for which

$$M = \sum_{\ell=1}^{r} a_\ell a_\ell^* \otimes b_\ell b_\ell^*, \tag{8}$$

where a^* denotes the complex conjugate of a, and $\otimes$ denotes the tensor product. We will not go into the details, but it suffices to think of separable states as fully explained by classical physics, in contradistinction, non-separable states, a.k.a. *entangled states* have special properties of interest in quantum physics. For rank-one states, i.e., if $\operatorname{rank}(M) = 1$, also called *pure states*, one can obtain a separable factorization by using *singular value decomposition (SVD)*. Non-rank-one states are called *mixed states*, and deciding whether a mixed state M is separable is, in general, NP-hard, see [28, 23].

Example of an entangled state [8]

Consider the following mixed state of size 9×9, hence $m = 3$. We have omitted showing zeros for readability, and we draw grid lines in order to highlight the block structure.

$$M = \left[\begin{array}{ccc|ccc|ccc} 1 & & & 1 & & & & & 1 \\ & 2 & & 1 & & & & & \\ & & \frac{1}{2} & & & & 1 & & \\ \hline & 1 & & \frac{1}{2} & & & & & \\ 1 & & & & 1 & & & & 1 \\ & & & & & 2 & & 1 & \\ \hline & & 1 & & & & 2 & & \\ & & & & & 1 & & \frac{1}{2} & \\ 1 & & & & 1 & & & & 1 \end{array}\right].$$

In [8] it is shown that M is entangled.

Analogously to other matrix ranks we considered thus far, there is also a notion of *separable rank*, see [13], sometimes called *optimal ensemble cardinality* [16], which we define for a separable matrix M as

$$\operatorname{rank}_{\text{sep}}(M) = \min\{r \in \mathbb{N} : M = \sum_{\ell=1}^{r} a_\ell a_\ell^* \otimes b_\ell b_\ell^* \text{ for some } a_\ell \text{ and } b_\ell \text{ in } \mathbb{C}^m\}. \quad (9)$$

A possible interpretation of the separable rank is that it gives a sense of how complex a classical system is, with the convention being that an entangled state has infinite separable rank. To our knowledge, the complexity of computing the separable rank is still unknown. There are some crude bounds on the separable rank though

$$\operatorname{rank}(M) \leq \operatorname{rank}_{\text{sep}}(M) \leq \operatorname{rank}(M)^2.$$

The left most inequality can be strict (see [16]) and the right most inequality follows from Caratheodory's theorem [41].

In addition to the above definition, there are several other variations on this notion of separability. One variation is to look for factorizations of the form $M = \sum_{\ell=1}^{r} A_\ell \otimes B_\ell$, where $A_\ell, B_\ell \in \mathcal{H}^m$ are Hermitian PSD matrices. From this it is easy to define the associated *mixed separable rank* as the smallest r for which such a factorization is possible. When M is diagonal its mixed separable rank equals the nonnegative rank of an associated $m \times m$ matrix consisting of the diagonal entries of M, see [14]. This shows that mixed separable rank is hard to compute.

Nonnegative tensor factorization ranks

Tensors, also called multi-way arrays, are natural generalizations of matrices commonly encountered in applied fields such as statistics, computer vision, and data science. Strictly speaking, tensor factorization ranks falls beyond the scope of this chapter, but given the similarities, we would be remiss not to include some remarks and references on the matter.

Consider, for example, a three-way array $T \in \mathbb{R}^{n\times m\times p}$. Then, its *tensor rank* is the smallest number $r \in \mathbb{N}$ of rank-one tensors (tensors of the form $a \otimes b \otimes c$ for some $a \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^p$) necessary to describe T, i.e.

$$\operatorname{rank}_{\text{tensor}}(T) = \min\{r \in \mathbb{N} : T = \sum_{\ell=1}^{r} a_\ell \otimes b_\ell \otimes c_\ell,\ a_\ell \in \mathbb{R}^n,\ b_\ell \in \mathbb{R}^m,\ c_\ell \in \mathbb{R}^p\}.$$

Similarly, one can define the *nonnegative tensor rank* by requiring the factors a_ℓ, b_ℓ, and c_ℓ to be nonnegative. Moreover, one can define the *symmetric tensor rank* by requiring $n = m = p$ and the factors to be equal, i.e., $a_\ell = b_\ell = c_\ell$ for all $\ell \in [r]$. An interesting effect of going to tensors is that some decompositions become unique [40]. See [9] for an applications-centric monograph on tensor factorization. For a mathematical survey, see Kolda and Bader [31].

Non-commutative matrix factorization ranks

We conclude this section with two more factorizations, often called non-commutative analogs of nonnegative- and CP factorizations. First is the *positive semi-definite (PSD) factorization*, where given a matrix $M \in \mathbb{R}_+^{m\times n}$ we look for PSD matrices $A_1, ..., A_m, B_1, ..., B_n \in \mathcal{S}_+^r$, for some $r \in \mathbb{N}$, such that the matrix M is described entry-wise as follows: $M_{i,j} = \langle A_i, B_j\rangle \in \mathbb{R}$ for $i \in [m]$ and $j \in [n]$. If the matrices A_i and B_j are diagonal for $i \in [m]$ and $j \in [n]$, then we recover a nonnegative factorization. Similar to nonnegative factorization, there is a substantial research interest in PSD-factorization, largely due to its many appealing geometric interpretations, including semi-definite representations of polyhedra. We refer the reader to the survey by Fawzi et al. [19] for further study of PSD-factorizations.

Second, we have the symmetric analog of PSD-factorization, called a *completely positive semi-definite (CPSD) factorization*, which simply adds the requirement that $n = m$ and $B_i = A_i$ for all $i \in [m]$.

The associated definition of rank accompanying these two factorizations should be clear. We stop introducing matrix factorization ranks now and shift gears towards proving bounds.

2 Bounding matrix factorization ranks

There are two modes of approximating factorization ranks. The first is from above, using heuristics to construct factorizations. The second is from below, via computing parameters, often combinatorial in nature, exploiting the support graph of the matrix M. The approach we follow falls in this latter category. We will explain the method applied to the CP-rank, though it should be clear what substitutions are needed to generalize it to the other factorization ranks we described in section 1.

In this section, we give the focal point of this chapter, the *moment hierarchy*. This is the core technique for approximating factorization ranks. In order to define the hierarchy, we start in section 2.1 with polynomial optimization problems, which we feel is more natural than jumping straight into generalized moment problems considered in section 2.2. Finally, we apply the tools built in section 2.2 to the setting of CP-rank in section 2.3.

The significant results concerning the properties of the hierarchy, like convergence, flatness, and exactness, will be mentioned and explained as we proceed. See the works [26, 27] for a more fleshed-out exposition of the process we follow here.

2.1 A brief introduction to polynomial optimization

We will be drawing heavily from the rich field of polynomial optimization, so it is only natural that we introduce some tools and notations in this regard. This small subsection is not an overview of the field. For that, we recommend the excellent works [34, 33]. We attempt to cover only the necessities needed to motivate the title and ease the reader into the more advanced machinery. Depending on the book's other chapters and the reader's background, some topics may be familiar, in which case, perusing this subsection will at least clarify the notation we use.

Consider the following optimization problem:

$$f^{\min} := \inf_{x \in K} f(x), \tag{10}$$

where

$$K := \{x \in \mathbb{R}^m : g_i(x) \geq 0 \ (i \in [p]) \ , \ h_j(x) = 0 \ (j \in [q])\}, \tag{11}$$

and $f, g_1, ..., g_p, h_1, ..., h_q \in \mathbb{R}[x]$ are polynomials in m variables $x_1, ..., x_m$. The domain of optimization, K, is a basic closed semi-algebraic set. Problem (10) is called a *polynomial optimization problem* or a *POP* for short. POPs are versatile tools for modeling various problems. Clearly, linear and quadratic programs are instances of POPs. Furthermore, one can encode binary variables with polynomial constraints of the form $x_i(x_i - 1) = 0$. Hence, many NP-hard problems can be reformulated as a POP, and, as such, POPs are generally hard to solve, see [33].

The moment approach to attacking a POP of the form (10) is as follows. We optimize the integral of the objective over probability measures that have support on

K, i.e., we consider the following problem

$$\begin{aligned} \text{val}^{\text{POP}} := \inf_{\mu \in \mathscr{M}(K)} & \int f(x) d\mu, \\ \text{s.t.} & \int 1 d\mu = 1, \end{aligned} \tag{12}$$

where $\mathscr{M}(K)$ is the set of all Borel measures supported on the set K.

Problems (10) and (12) are equivalent in the sense that they have the same optimal values, i.e., $f^{\min} = \text{val}^{\text{POP}}$. To see that $f^{\min} \geq \text{val}^{\text{POP}}$ holds consider the Dirac delta measure $\delta_{x^{\min}}$ supported at a minimizer $x^{\min}$ of problem (10). Then we have

$$\text{val}^{\text{POP}} \leq \int f(x) d\delta_{x^{\min}} = f(x^{\min}) = f^{\min}.$$

For the other inequality, $\text{val}^{\text{POP}} \geq f^{\min}$, simply observe

$$\int f(x) d\mu \geq f^{\min} \int 1 d\mu = f^{\min}.$$

The last equality comes from the fact that μ is a probability measure. With the equivalence between POPs and this new problem (12) established, we can focus on solving the latter.

2.2 Generalized moment problems

Problem (12) is a special instance of what is called a *generalized moment problem* (GMP), which is an even more versatile type of problem than a POP. Reformulating our optimization problem over measures does not give a clear advantage, as measures are difficult to handle. However, we will soon see in (16) how one can truncate the problem to create a hierarchy of *semi-definite programs* (SDP).

Consider the following general form of GMP:

$$\text{val} := \inf_{\mu \in \mathscr{M}(K)} \Big\{ \int f_0(x) d\mu : \int f_i d\mu = a_i \ (i \in [N]) \Big\}, \tag{13}$$

where $f_0, f_1, ..., f_N$ are polynomials. From the discussion in section 2.1, we saw that POPs are a special class of GMPs with $N = 1$ and $f_1 = a_1 = 1$. In section 2.3, we will show that the CP-rank can be reformulated as a GMP. Before we can start attacking the above GMP with the so-called *moment method*, we must first set some notation and basic definitions. Let $\mathbb{N}^m_t$ be the set of all *multi-indices* $\alpha = (\alpha_1, \alpha_2, ..., \alpha_m)$ such that $|\alpha| := \sum_{i=1}^m \alpha_i \leq t$. If $t = \infty$ we just write $\mathbb{N}^m$. For $x = (x_1, x_2, ..., x_m) \in \mathbb{R}^m$ denote the truncated sequence of monomials by $[x]_t := \left(x^\alpha\right)_{\alpha \in \mathbb{N}^m_t}$, where $x^\alpha := x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_m^{\alpha_m}$. For a measure μ define its *moments* to be the sequence of values, obtained when integrating the monomials w.r.t. the

measure, i.e,

$$\int x^\alpha d\mu \ \ (\alpha \in \mathbb{N}^m).$$

Using moments, we can think of measures as *linear functionals* acting on the ring of polynomials. That is, for a measure μ, we can define a linear map $L : \mathbb{R}[x] \to \mathbb{R}$ by defining what it does to monomials: $L(x^\alpha) = \int x^\alpha d\mu$ for every $\alpha \in \mathbb{N}^n$. Hence, for any polynomial $f = \sum_\alpha c_\alpha x^\alpha$, we have

$$L(f) = L(\sum_\alpha c_\alpha x^\alpha) = \sum_\alpha c_\alpha L(x^\alpha) = \sum_\alpha c_\alpha \int x^\alpha d\mu = \int f d\mu.$$

Denote the space of truncated linear functionals acting on the space of polynomials of degree at most t, i.e., on $\mathbb{R}[x]_t$, by $\mathbb{R}[x]_t^*$. Going in the opposite direction, i.e., from a linear functional $L \in \mathbb{R}[x]^*$ to a measure μ, is not always possible. When there does exists a measure $\mu \in \mathscr{M}(K)$ such that $L(x^\alpha) = \int x^\alpha d\mu$ for all $\alpha \in \mathbb{N}^m$, L is said to have a *representing measure*. We introduce some more concepts and notation to characterize the necessary conditions for measure representable functionals.

Recall the definition of our semi-algebraic set K in eq. (11):

$$K := \{x \in \mathbb{R}^m : g_i(x) \geq 0 \ (i \in [p]) \ , \ h_j(x) = 0 \ (j \in [q])\}.$$

For $t \in \mathbb{N} \cup \{\infty\}$ define the *truncated quadratic module* generated by $\mathbf{g} := (g_0, g_1, g_2, ..., g_p)$, with $g_0 = 1$, as

$$\mathcal{M}(\mathbf{g})_{2t} := \Big\{ \sum_{j=0}^{p} \sigma_j g_j : \sigma_j \in \Sigma, \ \deg(\sigma_j g_j) \leq 2t \Big\}. \tag{14}$$

Here Σ denotes the set of *sums of squares of polynomials*. In a similar vein we define the truncated *ideal* generated by $\mathbf{h} := (h_1, h_2, ..., h_q)$ as

$$\mathcal{I}(\mathbf{h})_{2t} := \Big\{ \sum_{j=1}^{q} \gamma_j h_j : \gamma_j \in \mathbb{R}[x], \ \deg(\gamma_j h_j) \leq 2t \Big\}. \tag{15}$$

When $t = \infty$ we also just drop the subscript and write: $\mathcal{M}(\mathbf{g})$ and $\mathcal{I}(\mathbf{h})$.

A crucial observation (see [33]) is that if $L \in \mathbb{R}[x]^*$ has a representing measure $\mu \in \mathscr{M}(K)$ then $L \geq 0$ on $\mathcal{M}(\mathbf{g})$ and $L = 0$ on $\mathcal{I}(\mathbf{h})$. Using this, we can define, for any $t \in \mathbb{N} \cap \{\infty\}$, the following sequence of parameters,

$$\begin{aligned} \xi_t := \min\{L(f_0) : & L \in \mathbb{R}[x]_{2t}^*, \\ & L(f_i) = a_i \ \ (i \in [N]), \\ & L \geq 0 \text{ on } \mathcal{M}(\mathbf{g})_{2t}, \\ & L = 0 \text{ on } \mathcal{I}(\mathbf{h})_{2t}\}. \end{aligned} \tag{16}$$

We call $\xi_1, \xi_2, ..., \xi_\infty$ a hierarchy as we clearly have for any $t \in \mathbb{N}$ that,

$$\xi_t \leq \xi_{t+1} \leq \xi_\infty \leq \text{val}.$$

Exercise 1

Given a solution L to the problem associated with ξ_{t+1} construct a solution to the problem associated with ξ_t.

Under mild assumptions, the bounds ξ_t converge asymptotically to the optimum value val as t goes to infinity. We state the well-known and widely used result here and refer to [33, 12] for a full exposition.

Theorem 1 *Assume problem (13) is feasible and the following Slater-type condition holds:*

$$\textit{there exist scalars } z_0, z_1, \ldots, z_N \in \mathbb{R} \textit{ such that } \sum_{i=0}^{N} z_i f_i(x) > 0 \textit{ for all } x \in K.$$

Then (13) has an optimal solution μ, which can be chosen to be finite atomic*, i.e., $\mu = \sum_{j\in J} c_j \delta_{x^{(j)}}$ for some finite index set J, scalars $c_j \geq 0$, and points $x^{(j)} \in K$. If, in addition, $\mathcal{M}(\mathbf{g})$ is Archimedean, i.e., $R - \sum_{i=1}^{m} x_i^2 \in \mathcal{M}(\mathbf{g})$ for some scalar $R > 0$, then we have* $\lim_{t\to\infty} \xi_t = \xi_\infty = \text{val}$.

Hence, the link between the GMP (13) and the hierarchy (16) is established.

Earlier, we said that, for each t, problem (16) is an SDP. This fact may become apparent after the following characterizations. Firstly, observe that a polynomial $\sigma \in \mathbb{R}[x]_{2t}$ is a sum of squares, i.e. $\sigma \in \Sigma$, if and only if there exists some matrix $M_\sigma \succeq 0$ such that $\sigma(x) = [x]_t^T M_\sigma [x]_t$. Having this in mind, we see that $L \geq 0$ on Σ_{2t} is equivalent to saying $L([x]_t[x]_t^T) \succeq 0$, because $L(\sigma) = L([x]_t^T M_\sigma [x]_t) = \langle L([x]_t[x]_t^T), M_\sigma \rangle$, and using the fact that the PSD cone is self-dual. Similarly, for any $\sigma \in \Sigma_{2t}$ and $j \in [p]$ we have $L(g_j\sigma) = \langle L(g_j(x)[x]_t[x]_t^T), M_\sigma \rangle$. Thus $L \geq 0$ on $\mathcal{M}(\mathbf{g})_{2t}$ can be equivalently characterized by the PSD constraints:

$$L(g_j [x]_{t-d_{g_j}} [x]_{t-d_{g_j}}^T) \succeq 0, \tag{17}$$

for $j = 0, 1, ..., p$, where $d_{g_j} := \lceil \deg(g_j)/2 \rceil$. Secondly, the ideal constraints $L = 0$ on $\mathcal{I}(\mathbf{h})_{2t}$ can be encoded as follows:

$$L(h_j [x]_{2t-\deg(h_j)}) = 0, \tag{18}$$

for each $j \in [q]$, where the vector equality should be understood entry-wise.

Exercise 2

Prove the above two claimed equivalences.

Using eq. (17) and eq. (18) we can reformulate problem (16) as

$$\begin{aligned} \xi_t = \min\{L(f) : & L \in \mathbb{R}[x]_{2t}^*, \\ & L(f_i) = a_i \ \ (i \in [N]), \\ & L(g_j[x]_{t-d_{g_j}}[x]_{t-d_{g_j}}^T) \succeq 0 \ (j = 0, 1, ..., p), \\ & L(h_j[x]_{2t-\deg(h_j)}) = 0 \ (j \in [q])\}. \end{aligned} \tag{19}$$

For fixed level t, the program (19) is an SDP of size polynomial in m. It is known that SDPs are efficiently solvable under some technical conditions, see [36]. However, computing val remains inefficient because the matrices describing (19) could be of size $\max_{j\in[p]}\{\binom{m+t-d_{g_j}}{t-d_{g_j}}\}$, and hence soon grow beyond what most currently available hardware can store in memory. In our experience the level t of the hierarchy is often quite small (1 to 5) for practical examples.

Finite convergence and recovering optimal solutions

Thus far in this section, we have seen how to get successive approximations of val. We saw in the preceding section 2.1 how GMPs relate to POPs, and soon in section 2.3 we will see how GMPs relate to the CP-rank of a matrix. However, what was not shown is whether we can also recover an optimizer $x^{\min}$. On another note, we said that the hierarchy quickly exceeds hardware capacity as the level increase, so it would be helpful if we had finite convergence, i.e., $\xi_t = \xi_\infty$ for some $t < \infty$. It turns out that there is a condition under which we solve both of these shortcomings mentioned above: finite convergence and the possibility of recovering an optimizer. Simply put, if a solution L to the hierarchy ξ_t at level t satisfies the *flatness condition* (20), then the bound at that level is exact, i.e., $\xi_t = \xi_\infty$, and there is a way to extract a finite atomic solution to the GMP in eq. (13). We formally state the classical theorem due to Curto and Fialkow [10, 11], upon which we base these claims. Note that the original formulation by Curto and Fialkow was not in the context of GMPs. We refer the reader again to [33, 12] for a more cohesive view.

Theorem 2 ***(Flatness theorem)*** *[10, 11] Consider the set K from (11) and define $d_K := \max\{1, \lceil \deg(g_j)/2 \rceil : j \in [p]\}$. Let $t \in \mathbb{N}$ such that $2t \geq \max\{\deg(f_i) : 0 \leq i \leq N\}$ and $t \geq d_K$. Assume $L \in \mathbb{R}[x]_{2t}^*$ is an optimal solution to the program (19) defining the parameter ξ_t and it satisfies the following* flatness condition*:*

$$\operatorname{rank} L([x]_s[x]_s^T) = \operatorname{rank} L([x]_{s-d_K}[x]_{s-d_K}^T) \text{ for some integer } s \text{ such that } d_K \leq s \leq t. \tag{20}$$

Then equality $\xi_t = \text{val}$ holds and problem (13) has an optimal solution μ which is finite atomic and supported on $r := \operatorname{rank} L([x]_{s-d_K}[x]_{s-d_K}^T)$ points in K.

For the details on how to extract the atoms of the optimal measure when the flatness condition of Theorem 2 holds, we refer to [30, 34].

In the context of factorization, we will soon see that the convergence described in Theorems 1 and 2 is not towards the rank but instead another closely related convex parameter. Furthermore, for the cases of nonnegative- and CP factorization, the atoms we recover are exactly the columns of the factorization matrices. We now proceed to apply the above techniques to the task of approximating the completely positive rank.

2.3 Constructing a hierarchy of lower bounds for CP-rank

With the moment hierarchy machinery in place, we return our attention to factorization ranks. In particular, we will construct a hierarchy of lower bounds for the CP-rank. It should be clear to the reader how to extend the contents of this section to the nonnegative and separable ranks. As for the other ranks discussed in section 1, some technicalities will be required, which we omit for brevity and simply provide references where appropriate.

Begin by recalling the definition of the CP-rank from eq. (6), and note that we can equivalently express it as follows:

$$\operatorname{rank}_{\mathrm{cp}}(M) = \min\{r \in \mathbb{N} : M = \sum_{\ell=1}^{r} a_\ell a_\ell^T \text{ for some } a_1, a_2, ..., a_r \in \mathbb{R}^m_+\}. \quad (21)$$

The above is sometimes called an *atomic* formulation because the factors a_ℓ (the columns of A in the eq. (6) formulation) can be thought of as the atoms of the factorization. Assuming we know that M is CP, we believe solving the optimization problem (21) is still hard, though, to the best of our knowledge, there is no proof of this claim. It is natural to ask if relaxing some constraints yields an easier problem. In this vein, Fawzi and Parrilo [21] introduced a natural "convexification" of the CP-rank:

$$\tau_{\mathrm{cp}}(M) := \inf\left\{\lambda : \frac{1}{\lambda} M \in \operatorname{conv}\{xx^T : x \in \mathbb{R}^m_+,\ M - xx^T \geq 0,\ M \geq xx^T\}\right\}. \quad (22)$$

A similar parameter can be defined for the NN-rank [21] and the separable rank [27].

Exercise 3

Prove that $\tau_{\mathrm{cp}}(M)$ is a lower bound for $\operatorname{rank}_{\mathrm{cp}}(M)$.

Because τ_{cp} is a convex relaxation of the combinatorial parameter $\operatorname{rank}_{\mathrm{cp}}$, it is possibly strictly worse, i.e., $\tau_{\mathrm{cp}}(M) < \operatorname{rank}_{\mathrm{cp}}(M)$ for some M. Furthermore, $\tau_{\mathrm{cp}}(M)$ does not appear any easier to compute than $\operatorname{rank}_{\mathrm{cp}}(M)$, in part because we do not have an efficient characterization of the convex hull described in eq. (22). However, not all is lost, as $\tau_{\mathrm{cp}}(M)$ can be reformulated as a GMP,

$$\tau_{\text{cp}}(M) = \inf_{\mu \in \mathcal{M}(K^M)} \left\{ \int_{K^M} 1 d\mu : \int_{K^M} x_i x_j d\mu = M_{ij} \ (i,j \in [m]), \right\}, \tag{23}$$

where

$$\begin{aligned} K^M := \{x \in \mathbb{R}^m : & \sqrt{M_{ii}} x_i - x_i^2 \geq 0 \ \ (i \in [m]), \\ & M_{ij} - x_i x_j \geq 0 \ \ (i \neq j \in [m]), \\ & M - xx^T \succeq 0\}. \end{aligned} \tag{24}$$

For a small proof using Theorem 1 see Lemma 2 of [32]. The idea of using a GMP to model CP matrices was already explored in the work of Nie [37].

The reader may wonder why the constraints $M_{ij} - x_i x_j \geq 0$ and $\sqrt{M_{ii}} x_i - x_i^2 \geq 0$ are preferred here over the equivalent and more intuitive constraints: $x_i \geq 0$ and $M_{ij} - x_i x_j \geq 0$ (for all $i, j \in [m]$). This is because the former gives, for finite t, a larger truncated quadratic module, which in turn gives better bounds for the finite levels of the hierarchy. Both options are, of course, equivalent in the limit as t goes to infinity, see [26].

Note that the last constraint, $M - xx^T \succeq 0$, is a polynomial matrix constraint. The idea behind this constraint is to encode that any CP factor aa^T of M is PSD less than M, i.e., $M - aa^T \succeq 0$. We could equivalently have asked that $f_v(x) := v^T(M - xx^T)v \geq 0$ for all $v \in \mathbb{R}^n$, or that the minors of $M - xx^T$ be nonnegative. However the matrix formulation is computationally easier to implement as we will see below. Several characterizations and supplementary references are considered for this constraint in [27].

Now we simply apply the techniques of section 2.1 to construct a hierarchy of lower bounds for the above GMP (23) to obtain the following parameter for any $t \in \mathbb{N} \cup \{\infty\}$:

$$\begin{aligned} \xi_t^{\text{cp}}(M) := \min\{L(1) : & L \in \mathbb{R}[x]_{2t}^*, \\ & L(xx^T) = M, \\ & L([x]_t [x]_t^T) \succeq 0, \\ & L((\sqrt{M_{ii}} x_i - x_i^2)[x]_{t-1}[x]_{t-1}^T) \geq 0 \ (i \in [m]), \\ & L((M_{ij} - x_i x_j)[x]_{t-1}[x]_{t-1}^T) \geq 0 \ (i \neq j \in [m]), \\ & L((M - xx^T) \otimes [x]_{t-1}[x]_{t-1}^T) \geq 0\}. \end{aligned} \tag{25}$$

The basic idea for the construction of this hierarchy comes initially from [26]. The last constraint was added later in [27].

Using Theorem 1 we have the following chain of inequalities:

$$\xi_1^{\text{cp}}(M) \leq \xi_2^{\text{cp}}(M) \leq \ldots \leq \xi_\infty^{\text{cp}}(M) = \tau_{\text{cp}}(M) \leq \text{rank}_{\text{cp}}(M).$$

Let us get some intuition for why this hierarchy works. Consider a CP factorization $a_1, \ldots, a_r \in \mathbb{R}_+^m$ of M with $r := \text{rank}_{\text{cp}}(M)$. Define for each $i \in [r]$ the following

evaluation linear functional L_{a_i} that maps a polynomial $f(x)$ to its evaluation at the point a_i, i.e.,

$$L_{a_i} : \mathbb{R}[x] \ni f(x) \mapsto f(a_i) \in \mathbb{R} \ \ (i \in [r]).$$

Then $\tilde{L} := \sum_{i\in[r]} L_{a_i}$ is feasible for the problem (25). Moreover, $\tilde{L}(1) = r = \text{rank}_{\text{cp}}(M)$.

Exercise 4

Show that $\tilde{L}$ satisfies each of the constraints of problem (25).

One can think of the constraints in (25) as filters excluding solutions that are dissimilar to $\tilde{L}$. Of course, $\tilde{L}$ is only feasible and not necessarily optimal. Hence $\xi_t(M) \leq \text{rank}_{\text{cp}}(M)$ for every t, and in practice, the inequality is often strict. The finite atomic measure $\tilde{\mu} := \sum_{i=1}^{r} \delta_{a_i}$ supported on the atoms $a_1, ..., a_r$ is a representing measure of $\tilde{L}$.

With a hierarchy constructed, we can now compute some examples.

2.4 A note on computing hierarchies of SDPs

We said before that, for a fixed $t \in \mathbb{N}$, problem (25) is an SDP, and we claimed that it could be computed efficiently. We now give some tips in implementing these problems using freely available software. Theory is often a poor substitute for hands-on experience when it comes to implementing code. Therefore, this small subsection is simply a snapshot of the quickly changing available tools. The reader is encouraged to play around with these tools should he/she seek a deeper understanding. At the end, we list a table of results so that the reader can get a feel for the power of these approximation hierarchies.

Disclaimer, the procedure we describe here is based on the author's experience and preferences and should not be seen as the only way to compute hierarchies.

The core idea is to work inside the programming language Julia [3], within which there is a package called JuMP [18] specially designed as a high-level interface between several commonly used solvers and Julia. In particular, JuMP can interface with MOSEK [1], a powerful commercial *interior-point solver*. Though MOSEK requires a license to run, academic licenses are available free of charge at the time of writing this. To summarize, one installs Julia, imports JuMP, uses the JuMP syntax to formulate the desired SDP as a JuMP-model, and then one passes off the JuMP-model to MOSEK to be solved. In broad strokes, this was the procedure followed in [26, 27, 32] to compute bounds for the NN-, CP-, and separable ranks. Some code is available as a package [2].

[2] See the code repository: https://github.com/JAndriesJ/ju-cp-rank

Some numerical results for CP-rank

In [27], the hierarchy (25) was tested on several matrices with high CP-rank. See [4] for the construction and definitions of these matrices. We already saw $\widetilde{M}_7$ above in eq. (7). We now list the bound at level $t = 3$ of the hierarchy (25) for some of the other matrices in [4].

Table 1 Bounds for completely positive rank at level $t = 3$.

M	$\text{rank}(M)$	m	$\lfloor \frac{m^2}{4} \rfloor$	$\xi_3^{\text{cp}}(M)$	$\text{rank}_{\text{cp}}(M)$
M_7	7	7	12	11.4	14
$\widetilde{M}_7$	7	7	12	10.5	14
$\widetilde{M}_8$	8	8	16	14.5	18
$\widetilde{M}_9$	9	9	20	18.4	26

Level $t = 3$ was the highest that could be computed on the available hardware.

3 Exploiting sparsity

In this penultimate section, we explore *sparsity*, by which we mean the exploitation of zeros in the matrix M to obtain better bounds or faster computations. In particular, we will explore a special kind of sparsity called *ideal sparsity*, as defined in [32]. Recall in problem (16) that the ideal constraint in the SDP forces the measure to vanish on certain polynomials. In this section, we show how the zero entries in a matrix lead the measure vanish on particular monomials. Using this, we can replace the original measure by multiple measures, each with a smaller support than the original one. The motivation behind this divide-and-conquer tactic is that the measures with smaller support lead to SDPs with smaller matrices and hence are easier to compute. Surprisingly, the sparse hierarchy, which we will define in (33), is also stronger than its dense analog (16), in contradistinction to other sparsity techniques where one often sacrifices the quality of the bounds in favor of computational benefits.

We first begin with a general introduction to ideal sparsity for the GMP setting in section 3.1. With the basic idea established, we apply ideal sparsity to CP-rank in section 3.2 and construct a sparse analog to the hierarchy (25). Finally, we conclude this section with some results in section 3.3 demonstrating the benefits of this sparse hierarchy over its dense analog.

3.1 An abbreviated introduction to ideal sparsity

Let $V := [m]$, $E \subseteq \{\{i,j\} \in V \times V : i \neq j\}$, and let $\overline{E} := \{\{i,j\} \in V \times V : \{i,j\} \notin E,\ i \neq j\}$ be its complement. Suppose now that the semi-algebraic set from eq. (11) is defined as follows

$$K_E := \{x \in \mathbb{R}^m : g_i(x) \geq 0\ (i \in [p])\ ,\ x_i x_j = 0\ (\{i,j\} \in \overline{E})\}.$$

By definition, the ideal in eq. (15) becomes

$$\mathcal{I}_{E,2t} := \Big\{ \sum_{\{i,j\} \in \overline{E}} \gamma_{ij} x_i x_j : \gamma_{ij} \in \mathbb{R}[x]_{2t-2} \Big\} \subseteq \mathbb{R}[x]_{2t}. \tag{26}$$

Observe that we have $K_E \subseteq \mathcal{I}_E$. We plan to partition K_E in a particular way to eliminate the need for ideal constraints in the subsequent levels of the hierarchy.

Consider the undirected graph $G = (V, E)$ and denote its maximal cliques by $V_1, \ldots, V_s$. For each $k \in [s]$ define the following subset of K_E :

$$\widehat{K_k} := \{x \in K : \mathrm{supp}(x) \subseteq V_k\} \subseteq K \subseteq \mathbb{R}^m. \tag{27}$$

Here, $\mathrm{supp}(x) = \{i \in [m] : x_i \neq 0\}$ denotes the *support* of $x \in \mathbb{R}^m$. Observe that the sets $\widehat{K_1}, \ldots, \widehat{K_s}$ cover the set K_E:

$$K_E = \widehat{K_1} \cup \ldots \cup \widehat{K_s}. \tag{28}$$

It is an easy exercise to see that if $x \in K_E$, then its support $\mathrm{supp}(x)$ is a clique of the graph G, and thus it is contained in a maximal clique V_k, so that $x \in \widehat{K_k}$, for some $k \in [s]$. Now define $K_k \subseteq \mathbb{R}^{|V_k|}$ to be the projection of $\widehat{K_k}$ onto the subspace indexed by V_k:

$$K_k := \{y \in \mathbb{R}^{|V_k|} : (y, 0_{V \setminus V_k}) \in \widehat{K_k}\} \subseteq \mathbb{R}^{|V_k|}. \tag{29}$$

We use the notation $(y, 0_{V \setminus V_k})$ to denote the vector of $\mathbb{R}^n$ obtained from $y \in \mathbb{R}^{|V_k|}$ by padding it with zeros at all entries indexed by $V \setminus V_k$. For an n-variate function $f : \mathbb{R}^{|V|} \to \mathbb{R}$ and a subset $U \subseteq V$, we let $f_{|U} : \mathbb{R}^{|U|} \to \mathbb{R}$ denote the function in the variables $x(U) = \{x_i : i \in U\}$, which is obtained from f by setting to zero all the variables x_i indexed by $i \in V \setminus U$. That is, $f_{|V_k}(y) = f(y, 0_{V \setminus V_k})$ for $y \in \mathbb{R}^{|V_k|}$.

We may now define the following sparse analog of (13):

$$\mathrm{val}^{\mathrm{sp}} := \inf_{\mu_k \in \mathscr{M}(K_k), k \in [s]} \Big\{ \sum_{k=1}^{s} \int f_{0|V_k} d\mu_k \ : \sum_{k=1}^{s} \int f_{i|V_k} d\mu_k = a_i\ (i \in [N]) \Big\}. \tag{30}$$

Proposition 1 *[32] Problems (13) (using K_E) and (30) are equivalent, i.e., their optimum values are equal:* $\mathrm{val} = \mathrm{val}^{\mathrm{sp}}$.

Based on the reformulation (30) we can define the following *ideal-sparse* moment relaxation for problem (13): for any integer $t \in \mathbb{N} \cup \{\infty\}$

$$\xi_t^{\text{isp}} := \inf \Big\{ \sum_{k=1}^{s} L_k(f_{0|V_k}) : L_k \in \mathbb{R}[x(V_k)]_{2t}^{*} \ (k \in [s]), \\ \sum_{k=1}^{s} L_k(f_{i|V_k}) = a_i \ (i \in [N]), \\ L_k \geq 0 \text{ on } \mathcal{M}(\mathbf{g}_{|V_k})_{2t} \ (k \in [s]) \Big\}, \tag{31}$$

where $\mathbf{g}_{|V_k} := (g_{0|V_k}, g_{1|V_k}, g_{2|V_k}, ..., g_{p|V_k})$. Note that the ideal constraint are entirely captured by the fact that none of the measures μ_k support any elements of the ideal.

With two hierarchies converging to the same value, the obvious question is whether one converges faster. Surprisingly, the bounds for the sparse hierarchy (31) are at least as good as for the dense hierarchy (16).

Theorem 3 *[32] For any integer $t \in \mathbb{N} \cup \{\infty\}$, we have $\xi_t \leq \xi_t^{\text{isp}} \leq$ val. If, in addition $\mathcal{M}(\mathbf{g})$ is Archimedian and the condition in Theorem 1 holds, then $\lim_{t\to\infty} \xi_t^{\text{isp}} =$ val.*

The advantage of (31) over (16) is twofold. Firstly, the sparse bounds are at least as good as the dense bounds. Secondly, there is potential for computation speed-up since each set V_k can be much smaller than the whole set V. This holds despite there now being more variables and constraints overall. However, speed-up fails in cases where there are exponentially many (in n) maximal cliques, like when G is a complete graph with a perfect matching deleted.

Observe that chordality need not be assumed on the cliques. However, we are required to find all maximal cliques. For an arbitrary graph, this could be difficult, but in the setting of factorization ranks, the graphs are often small, with around 5 to 15 vertices. Hence, one can compute the maximal cliques using algorithms like the one described in [6].

3.2 Ideal sparsity in approximating CP-rank

Return now to the completely positive rank. The rather abstractly defined ideal constraint in section 3.1 will emerge naturally from the zeros in a matrix. Consider a CP matrix $M \in \mathcal{S}_+^m$, assume $M_{ii} > 0$ for all $i \in [m]$. If M is a CP matrix with $M_{ii} = 0$, then its i^{th} row and column are identically zero, and thus it can be removed without changing the CP-rank. Define the *support graph* $G_M := (V, E_M)$ of M, with edge-set and non-edge-set respectively defined by:

$$E_M := \{\{i, j\} : M_{ij} \neq 0,\ i, j \in V,\ i \neq j\},\ \overline{E}_M := \{\{i, j\} : M_{ij} = 0,\ i, j \in V,\ i \neq j\}.$$

If G_M is not connected, then M can be block-diagonalized using row and column permutations. It is immediately apparent that the CP-rank of a block diagonal matrix is the sum of the CP-ranks of its blocks. So we may assume that G_M is connected. Now we can modify the semi-algebraic set from eq. (24) to read as follows

$$\begin{aligned} K_M^{\mathrm{isp}} := \{x \in \mathbb{R}^m : & \sqrt{M_{ii}}x_i - x_i^2 \geq 0 \ (i \in [m]), \\ & M_{ij} - x_i x_j \geq 0 \ (\{i,j\} \in E_M), \\ & x_i x_j = 0 \ (\{i,j\} \in \overline{E}_M), \\ & M - xx^T \succeq 0\}. \end{aligned} \tag{32}$$

We have not introduced any new information. We have just explicitly encoded the fact that $M_{ij} = 0$ and $M_{ij} - x_i x_j \geq 0$ imply $x_i x_j = 0$ (because $x \geq 0$). In this form we can apply the results from sections 2.2, 2.3 and 3.1 to define the following new hierarchy,

$$\begin{aligned} \xi_t^{\mathrm{cp,isp}}(M) = \min \Big\{ \sum_{k=1}^{s} L_k(1) : & L_k \in \mathbb{R}[x(V_k)]_{2t}^* \ (k \in [s]), \\ & \sum_{k \in [s]: i,j \in V_k} L_k(x_i x_j) = M_{ij} \ (i,j \in V), \\ & L_k([x(V_k)]_t [x(V_k)]_t^T) \geq 0 \ (k \in [s]), \\ & L_k((\sqrt{M_{ii}}x_i - x_i^2)[x(V_k)]_{t-1}[x(V_k)]_{t-1}^T) \geq 0 \ (i \in V_k, \ k \in [s]), \\ & L_k((M_{ij} - x_i x_j)[x(V_k)]_{t-1}[x(V_k)]_{t-1}^T) \geq 0 \ (i \neq j \in V_k, \ k \in [s]), \\ & L_k((M - xx^T) \otimes [x(V_k)]_{t-1}[x(V_k)]_{t-1}^T) \geq 0, \ (k \in [s]). \end{aligned} \tag{33}$$

As a direct consequence of Theorem 3 we have the following relation:

$$\xi_t^{\mathrm{cp}}(M) \leq \xi_t^{\mathrm{cp,isp}}(M) \leq \tau_{\mathrm{cp}}(M).$$

Problem (33) looks cumbersome. However, it is ultimately just problem (25) with the single functional replaced by multiple functionals, each with support tailored to exclude polynomials in the ideal $\mathcal{I}_{E_M}$.

Observe that, if, in problem (33), we replace the matrix $M - xx^T$ by its principal submatrix indexed by V_k, then one also gets a lower bound on $\tau_{\mathrm{cp}}(M)$, at most $\xi_t^{\mathrm{cp,isp}}(M)$, but potentially cheaper to compute. We let $\xi_t^{\mathrm{cp,wisp}}(M)$ denote the parameter obtained in this way, by replacing in the definition of $\xi_t^{\mathrm{cp,isp}}(M)$ the last constraint by

$$L_k((M[V_k] - x(V_k)x(V_k)^T) \otimes [x(V_k)]_{t-1}[x(V_k)]_{t-1}^T) \geq 0 \ \text{ for } k \in [s], \tag{34}$$

so that we have

$$\xi_t^{\mathrm{cp,wisp}}(M) \leq \xi_t^{\mathrm{cp,isp}}(M).$$

An example of maximal cliques in the support graph of a matrix

To get some intuition into what the maximal cliques look like in the CP factorization setting, consider the following matrix and its associated support graph in fig. 1.

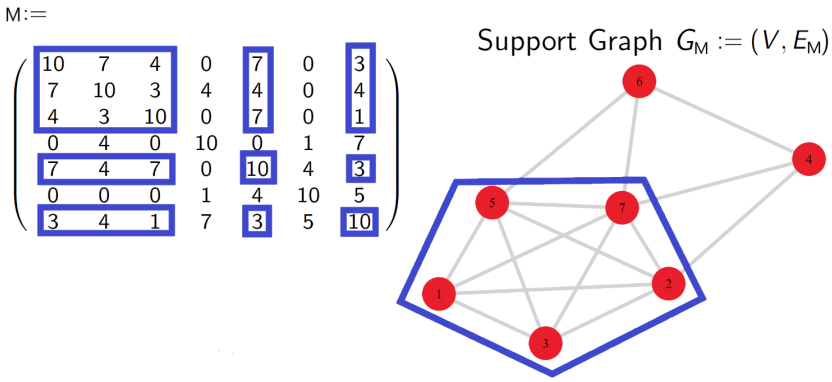

Fig. 1 Example of a matrix and its support graph. This example has non-edges: $\overline{E}_M := \{\{1,4\},\{1,6\},\{2,6\},\{3,4\},\{3,6\},\{4,5\}\}$, and maximal cliques: $V_1 := \{1,2,3,5,7\}$, $V_2 := \{2,4,7\}$, $V_3 := \{5,6,7\}$, $V_4 := \{4,6,7\}$. Hence, if M is CP, then its factors can only be supported by one of these four cliques.

3.3 Advantages of the sparse hierarchy

In this subsection, we compare the dense and sparse hierarchies for approximating the CP-rank. The comparison is first made in terms of bounds and then in terms of computational speed-up.

Better bounds

We now demonstrate some advantages of the sparse hierarchy (33) above its dense counterpart in (25). To this end consider one of the matrices from [5], namely,

$$\widehat{M} = \begin{bmatrix}
91 & 0 & 0 & 0 & 19 & 24 & 24 & 24 & 19 & 24 & 24 & 24 \\
0 & 42 & 0 & 0 & 24 & 6 & 6 & 6 & 24 & 6 & 6 & 6 \\
0 & 0 & 42 & 0 & 24 & 6 & 6 & 6 & 24 & 6 & 6 & 6 \\
0 & 0 & 0 & 42 & 24 & 6 & 6 & 6 & 24 & 6 & 6 & 6 \\
19 & 24 & 24 & 24 & 91 & 0 & 0 & 0 & 19 & 24 & 24 & 24 \\
24 & 6 & 6 & 6 & 0 & 42 & 0 & 0 & 24 & 6 & 6 & 6 \\
24 & 6 & 6 & 6 & 0 & 0 & 42 & 0 & 24 & 6 & 6 & 6 \\
24 & 6 & 6 & 6 & 0 & 0 & 0 & 42 & 24 & 6 & 6 & 6 \\
19 & 24 & 24 & 24 & 19 & 24 & 24 & 24 & 91 & 0 & 0 & 0 \\
24 & 6 & 6 & 6 & 24 & 6 & 6 & 6 & 0 & 42 & 0 & 0 \\
24 & 6 & 6 & 6 & 24 & 6 & 6 & 6 & 0 & 0 & 42 & 0 \\
24 & 6 & 6 & 6 & 24 & 6 & 6 & 6 & 0 & 0 & 0 & 42
\end{bmatrix}.$$

For this matrix we know that $\text{rank}_{\text{cp}}(\widehat{M}) = 37$. At the first level $t = 1$, we have $\xi_1^{\text{cp,isp}}(\widehat{M}) = 29.66$ while the dense hierarchy gives $\xi_1^{\text{cp}}(\widehat{M}) = 4.85$. Going to higher levels does improve the dense bound to $\xi_2^{\text{cp}}(\widehat{M}) = 29.66$, but the sparse bound does not seem to change.

In [32], it was shown that the separation between $\xi_1^{\text{cp,isp}}(M)$ and $\xi_1^{\text{cp}}(M)$ could be made arbitrarily big by taking matrices M of the form:

$$M = \begin{pmatrix} (m+1)I_m & J_m \\ J_m & (m+1)I_m \end{pmatrix} \in \mathcal{S}^{2m}$$

and increasing m. Here I_m is the identity matrix, and J_m is the all-ones matrix.

This gap is motivated by the sparse hierarchy incorporating certain structural information that the dense hierarchy ignores. To understand what we mean, consider first the *edge clique-cover number* $c(G)$ of the graph G defined to be the minimal number of cliques needed to cover all edges of G. In Lemma 13 of [32] it is shown that

$$\xi_1^{\text{cp,wisp}}(M) \geq c_{\text{frac}}(G_M),$$

where $c_{\text{frac}}(G)$ is the *fractional edge clique-cover number*, the natural linear relaxation of $c(G)$:

$$c_{\text{frac}}(G) := \min\Big\{ \sum_{k=1}^{s} x_k : \sum_{k:\{i,j\}\subseteq V_k} x_k \geq 1 \ \text{ for } \{i,j\} \in E \Big\}.$$

Exercise 5

Compute the fractional and usual (integer) clique-cover number of $G_{\widehat{M}}$.

Hence we have the following relations:

$$\begin{array}{ccccccccc}
c_{\text{frac}}(G_M) & \leq & \xi_1^{\text{cp,wisp}}(M) & \leq & \xi_2^{\text{cp,wisp}}(M) & \leq \ldots \leq & \xi_\infty^{\text{cp,wisp}}(M). & & \\
 & & \wedge\mid & & \wedge\mid & & \mid\mid & & \\
 & & \xi_1^{\text{cp,isp}}(M) & \leq & \xi_2^{\text{cp,isp}}(M) & \leq \ldots \leq & \xi_\infty^{\text{cp,isp}}(M) & = \tau_{\text{cp}}(M) & \leq \text{rank}_{\text{cp}}(M) \\
 & & \vee\mid & & \vee\mid & & \mid\mid & & \\
 & & \xi_1^{\text{cp}}(M) & \leq & \xi_2^{\text{cp}}(M) & \leq \ldots \leq & \xi_\infty^{\text{cp}}(M). & &
\end{array}$$

The weak sparse hierarchy $\xi_t^{\text{cp,wisp}}$ and the dense hierarchies ξ_t^{cp} are incomparable, as there are examples where one outperforms the other and vice versa.

Speed-up in computation

We said before that the sparse hierarchy involves smaller SDPs than the dense version and, as a result, can be computed faster.

To demonstrate this, the hierarchies are tested on a family of randomly generated CP-matrices ordered ascending in size and ascending in *nonzero density*. The nonzero density of a matrix M is the fraction of entries above the diagonal that are nonzero, hence for the identity matrix, it would be zero, and for a dense matrix with no zeros, it would be one. This parameter is crude in that it is oblivious to the structure of the support graph. Nonetheless, it suffices to show how the speed-up is related to the sparsity in the matrix. Consider the following fig. 2 taken from [32].

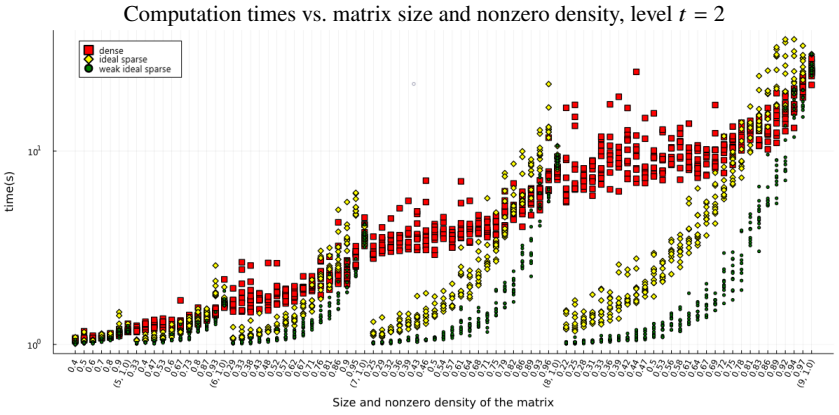

Fig. 2 Scatter plot of the computation times (in seconds) for the three hierarchies $\xi_{2,\dagger}^{\mathrm{cp}}$ (indicated by a red square), $\xi_{2,\dagger}^{\mathrm{cp,isp}}$ (indicated by a yellow lozenge), $\xi_{2,\dagger}^{\mathrm{cp,wisp}}$ (indicated by a green circle) against matrix size and nonzero density for 850 random matrices. The matrices are arranged in ascending size ($n = 5, 6, 7, 8, 9$) and then ascending nonzero density, ranging from the minimal density needed to have a connected support graph to a fully dense matrix. For each size and nonzero density ten examples were computed to account for different support graphs.

The hierarchies $\xi_{t,\dagger}^{\mathrm{cp}}$, $\xi_{t,\dagger}^{\mathrm{cp,isp}}$, and $\xi_{t,\dagger}^{\mathrm{cp,wisp}}$ are slight modifications of the familiar parameters ξ_t^{cp}, $\xi_t^{\mathrm{cp,isp}}$, and $\xi_t^{\mathrm{cp,wisp}}$ described already. The exact definition is avoided here because there are several technicalities to consider that will only detract from the core message, which is that the sparse hierarchy is potentially much faster when there are many zeros in the matrix.

4 Summary

Finally, we summarise this chapter. In section 1, we introduced the reader to several factorization ranks and motivated their importance with applications and links to other branches of science. After building the general tools needed, we focused on approximating the CP-rank in section 2. We then improved our approximation in section 3 by including structural information about the matrix support graph before demonstrating the improvement with theoretical and numerical results. We hope to have convinced the reader of the generality and utility of polynomial optimization techniques in dealing with the difficult and pertinent problem of matrix factorization.

Acknowledgements

We want to thank Prof. Dr. Monique Laurent for proofreading several drafts of this chapter and providing key insights when the author's knowledge was lacking. We also thank the editors for the opportunity to consolidate and share our expertise on this fascinating topic.

References

1. E. D. Andersen and K. D. Andersen. The MOSEK interior point optimizer for linear programming: An implementation of the homogeneous algorithm. H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, *High Performance Optimization*, 197–232. Springer US, Boston, MA, 2000.
2. A. Berman and N. Shaked-Monderer. *Completely Positive Matrices*. WORLD SCIENTIFIC, 2003.
3. J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.
4. I. M. Bomze, W. Schachinger, and R. Ullrich. From seven to eleven: Completely positive matrices with high cp-rank. *Linear Algebra and its Applications*, 459:208–221, 2014.
5. I. M. Bomze, W. Schachinger, and R. Ullrich. New lower bounds and asymptotics for the cp-rank. *SIAM Journal on Matrix Analysis and Applications*, 36(1):20–37, 2015.
6. C. Bron and J. Kerbosch. Algorithm 457: Finding all cliques of an undirected graph. *Communications of The Acm*, 16(9):575–577, Sept. 1973.
7. S. Burer. On the copositive representation of binary and continuous nonconvex quadratic programs. *Mathematical Programming*, 120(2):479–495, Sept. 2009.
8. M.-D. Choi. Positive linear maps. *Operator Algebras and Applications*, volume 38 of *Proc. Sympos. Pure Math.*, pages 583–590. 1982.
9. A. Cichocki, R. Zdunek, A. H. Phan, and S-i. Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*. Wiley Publishing, 2009.
10. R. E. Curto and L. A. Fialkow. Solution of the truncated complex moment problem for flat data. 1996.
11. R. E. Curto and L. A. Fialkow. The truncated complex K-moment problem. *Transactions of the American Mathematical Society*, 352:2825–2855, 2000.
12. E. de Klerk and M. Laurent. A survey of semidefinite programming approaches to the generalized problem of moments and their error analysis. *Association for Women in Mathematics Series*, 2019.

13. G. De las Cuevas, T. Drescher, and T. Netzer. Separability for mixed states with operator Schmidt rank two. *Quantum*, 3:203, Dec. 2019.
14. G. de las Cuevas and T. Netzer. Mixed states in one spatial dimension: Decompositions and correspondence with nonnegative matrices. *Journal of Mathematical Physics*, 2020.
15. P. J. C. Dickinson and L. Gijben. On the computational complexity of membership problems for the completely positive cone and its dual. *Computational Optimization and Applications*, 57:403–415, 2014.
16. D. P. Divincenzo, B. M. Terhal, and Ashish V. Thapliyal. Optimal decompositions of barely separable states. *Journal of Modern Optics*, 47(2-3):377–385, 2000.
17. J. H. Drew, C. R. Johnson, and R. Loewy. Completely positive matrices associated with M-matrices. *Linear & Multilinear Algebra*, 37:303–310, 1994.
18. I. Dunning, J. Huchette, and M. Lubin. JuMP: A modeling language for mathematical optimization. *Siam Review*, 59:295–320, 2017.
19. H. Fawzi, J. Gouveia, P. A. Parrilo, R. Z. Robinson, and R. R. Thomas. Positive semidefinite rank. *Mathematical Programming*, 153(1):133–177, Oct. 2015.
20. H. Fawzi, J. Gouveia, P. A. Parrilo, J. Saunderson, and R. R. Thomas. Lifting for simplicity: Concise descriptions of convex sets. *SIAM Review*, 64(4):866–918, 2022.
21. H. Fawzi and P. A. Parrilo. Self-scaled bounds for atomic cone ranks: Applications to nonnegative rank and cp-rank. *Mathematical Programming*, 158(1):417–465, July 2016.
22. S. Fiorini, S. Massar, S. Pokutta, H. R. Tiwary, and R. de Wolf. Exponential lower bounds for polytopes in combinatorial optimization. *Journal of The Acm*, 62(2), May 2015.
23. S. Gharibian. Strong NP-hardness of the quantum separability problem. *Quantum Information & Computation*, 10:343–360, 2010.
24. N. Gillis. *Nonnegative Matrix Factorization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2020.
25. N. Gillis and F. Glineur. On the geometric interpretation of the nonnegative rank. *Linear Algebra and its Applications*, 437(11):2685–2712, 2012.
26. S. Gribling, D. de Laat, and M. Laurent. Lower bounds on matrix factorization ranks via noncommutative polynomial optimization. *Foundations of Computational Mathematics*, 1–58, 2019.
27. S. Gribling, M. Laurent, and A. Steenkamp. Bounding the separable rank via polynomial optimization. *Linear Algebra and its Applications*, 2022.
28. L. Gurvits. Classical deterministic complexity of Edmonds' problem and quantum entanglement. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 10–19, New York, NY, USA, 2003. Association for Computing Machinery.
29. M. Hall. *Combinatorial Theory*. John Wiley & Sons, 1988.
30. D. Henrion and J.-B. Lasserre. Detecting global optimality and extracting solutions in GloptiPoly. D. Henrion and A. Garulli, editors, *Positive Polynomials in Control*, 293–310. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
31. T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.
32. M. Korda, M. Laurent, V. Magron, and A. Steenkamp. Exploiting ideal-sparsity in the generalized moment problem with application to matrix factorization ranks. ArXiv, +2023.
33. J. B. Lasserre. *Moments, Positive Polynomials and Their Applications*. IMPERIAL COLLEGE PRESS, 2009.
34. M. Laurent. Sums of squares, moment matrices and optimization over polynomials. In M. Putinar and S. Sullivant, editors, *Emerging Applications of Algebraic Geometry*, 157–270. Springer New York, New York, NY, 2009.
35. D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.
36. Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.
37. J. Nie. The $\mathcal{A}$-truncated K-moment problem. *Foundations of Computational Mathematics*, 14(6):1243–1276, Dec. 2014.

38. N. Shaked-Monderer and A. Berman. *Copositive and Completely Positive Matrices*. World Scientific Publishing.
39. N. Shaked-Monderer, I. M. Bomze, F. Jarre, and W. Schachinger. On the cp-rank and minimal cp factorizations of a completely positive matrix. *SIAM Journal on Matrix Analysis and Applications*, 34:355–368, 2013.
40. N. D. Sidiropoulos and R. Bro. On the uniqueness of multilinear decomposition of N-way arrays. *Journal of Chemometrics*, 14(3):229–239, 2000.
41. A. Uhlmann. Entropy and Optimal Decompositions of States Relative to a Maximal Commutative Subalgebra. *Open Systems & Information Dynamics*, 5(3):209–228, Sept. 1998.
42. S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 20:1364–1377, 2009.
43. M. Yannakakis. Expressing combinatorial optimization problems by linear programs. *STOC '88*, 1988.