FULL LENGTH PAPER

Tensor principal component analysis via convex optimization

Bo Jiang · Shiqian Ma · Shuzhong Zhang

Abstract This paper is concerned with the computation of the principal components for a general tensor, known as the tensor principal component analysis (PCA) problem. We show that the general tensor PCA problem is reducible to its special case where the tensor in question is super-symmetric with an even degree. In that case, the tensor can be embedded into a symmetric matrix. We prove that if the tensor is rank-one, then the embedded matrix must be rank-one too, and vice versa. The tensor PCA problem can thus be solved by means of matrix optimization under a rank-one constraint, for which we propose two solution methods: (1) imposing a nuclear norm penalty in the objective to enforce a low-rank solution; (2) relaxing the rank-one constraint by semidefinite programming. Interestingly, our experiments show that both methods can yield a rank-one solution for almost all the randomly generated instances, in which

B. Jiang
Research Center for Management Science and Data Analytics,
School of Information Management and Engineering,
Shanghai University of Finance and Economics, Shanghai 200433, China

S. Ma (✉)
Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong, Shatin, NT, Hong Kong
e-mail: sqma@se.cuhk.edu.hk

S. Zhang
Department of Industrial and Systems Engineering,
University of Minnesota, Minneapolis, MN 55455, USA

case solving the original tensor PCA problem to optimality. To further cope with the size of the resulting convex optimization models, we propose to use the alternating direction method of multipliers, which reduces significantly the computational efforts. Various extensions of the model are considered as well.

Keywords Tensor · Principal component analysis · Low rank · Nuclear norm · Semidefinite programming relaxation

Mathematics Subject Classification (2010) 15A69 · 15A03 · 62H25 · 90C22 · 15A18

1 Introduction

Principal component analysis (PCA) plays an important role in applications arising from data analysis, dimension reduction and bioinformatics etc. PCA finds a few linear combinations of the original variables. These linear combinations, which are called principal components (PCs), are orthogonal to each other and explain most of the variance of the data. PCs provide a powerful tool to compress data along the direction of maximum variance to reach the minimum information loss. Specifically, let $\xi = (\xi_1, \ldots, \xi_m)$ be an m-dimensional random vector. Then for a given data matrix $A \in \mathbb{R}^{m \times n}$ which consists of n samples of the m variables, finding the PC that explains the largest variance of the variables $(\xi_1, \ldots, \xi_m)$ corresponds to the following optimization problem:

$$\left(\lambda^*, x^*, y^*\right) := \min_{\lambda \in \mathbf{R}, x \in \mathbf{R}^m, y \in \mathbf{R}^n} \|A - \lambda x y^\top\|. \tag{1}$$

Problem (1) is well known to be reducible to computing the largest singular value (and corresponding singular vectors) of A, and can be equivalently formulated as:

$$\begin{aligned} \max_{x,y} \quad & \begin{pmatrix} x \\ y \end{pmatrix}^\top \begin{pmatrix} 0 & A \\ A^\top & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ \text{s.t.} \quad & \left\| \begin{pmatrix} x \\ y \end{pmatrix} \right\| = 1. \end{aligned} \tag{2}$$

Note that the optimal value and the optimal solution of Problem (2) correspond to the largest eigenvalue and the corresponding eigenvector of the symmetric matrix $\begin{pmatrix} 0 & A \\ A^\top & 0 \end{pmatrix}$.

Although the PCA and eigenvalue problem for matrix have been well studied in the literature, the research of PCA for tensors is still lacking. Nevertheless, the tensor PCA is of great importance in practice and has many applications in computer vision [56], diffusion Magnetic Resonance Imaging (MRI) [2,19,50], quantum entanglement problem [27], spectral hypergraph theory [30] and higher-order Markov chains [37]. This is mainly because in real life we often encounter multidimensional data, including images, video, range data and medical data such as CT and MRI. A color image can

be considered as 3D data with row, column, color in each direction, while a color video sequence can be considered as 4D data, where time is the fourth dimension. Moreover, it turns out that it is more reasonable to treat the multidimensional data as a tensor instead of unfolding it into a matrix. For example, Wang and Ahuja [56] reported that the images obtained by tensor PCA technique have higher quality than that by matrix PCA. Similar to its matrix counterpart, the problem of finding the PC that explains the most variance of a tensor $\mathcal{A}$ (with degree m) can be formulated as:

$$
\begin{aligned}
\min \quad & \|\mathcal{A} - \lambda x^1 \otimes x^2 \otimes \cdots \otimes x^m\| \\
\text{s.t.} \quad & \lambda \in \mathbf{R}, \ \|x^i\| = 1, i = 1, 2, \ldots, m,
\end{aligned}
\tag{3}
$$

which is equivalent to

$$
\begin{aligned}
\max \quad & \mathcal{A}(x^1, x^2, \ldots, x^m) \\
\text{s.t.} \quad & \|x^i\| = 1, i = 1, 2, \ldots, m,
\end{aligned}
\tag{4}
$$

where $\otimes$ denotes the outer product between vectors; viz.

$$
(x^1 \otimes x^2 \otimes \cdots \otimes x^m)_{i_1 i_2 \ldots i_m} = \prod_{k=1}^{m} (x^k)_{i_k}.
$$

Let us call the above solution the *leading* PC. Once the leading PC is found, the other PCs can be computed sequentially via the so-called "deflation" technique. For instance, the second PC is defined as the leading PC of the tensor subtracting the leading PC from the original tensor, and so forth. The theoretical basis of such deflation procedure for tensors is not exactly sound, although its matrix counterpart is well established (see [44] and the references therein for more details). However, the deflation process does provide a heuristic way to compute multiple PCs of a tensor, albeit approximately. Thus in the rest of this paper, we focus on finding the leading PC of a tensor.

Problem (4) is also known as the best rank-one approximation of tensor $\mathcal{A}$. As we shall see later, problem (4) can be reformulated as

$$
\begin{aligned}
\max \quad & \mathcal{F}(x, x, \ldots, x) \\
\text{s.t.} \quad & \|x\| = 1,
\end{aligned}
\tag{5}
$$

where $\mathcal{F}$ is a super-symmetric tensor. Problem (5) is NP-hard and is known as the maximum Z-eigenvalue problem. Note that a variety of eigenvalues and eigenvectors of a real symmetric tensor were introduced by Lim [38] and Qi [48] independently in 2005. Since then, various methods have been proposed to find the Z-eigenvalues [10,31–33,49], which however may correspond only to local optimums. In this paper, we shall focus on finding the global optimal solution of (5).

Before proceeding let us introduce notations that will be used throughout the paper. We denote $\mathbf{R}^n$ to be the n-dimensional Euclidean space. A tensor is a high dimensional array of real data, usually in calligraphic letter, and is denoted as $\mathcal{A} = (\mathcal{A}_{i_1 i_2 \ldots i_m})_{n_1 \times n_2 \times \cdots \times n_m}$. The space where $n_1 \times n_2 \times \cdots \times n_m$ dimensional

real-valued tensors reside is denoted by $\mathbf{R}^{n_1 \times n_2 \times \cdots \times n_m}$. We call $\mathcal{A}$ super-symmetric if $n_1 = n_2 = \cdots = n_m$ and $\mathcal{A}_{i_1 i_2 \ldots i_m}$ is invariant under any permutation of $(i_1, i_2, \ldots, i_m)$, i.e., $\mathcal{A}_{i_1 i_2 \ldots i_m} = \mathcal{A}_{\pi(i_1, i_2, \ldots, i_m)}$, where $\pi(i_1, i_2, \ldots, i_m)$ is any permutation of indices $(i_1, i_2, \ldots, i_m)$. The space where $\underbrace{n \times n \times \cdots \times n}_{m}$ super-symmetric tensors reside is denoted by $\mathbf{S}^{n^m}$. Special cases of tensors are vector ($m = 1$) and matrix ($m = 2$), and tensors can also be seen as a long vector or a specially arranged matrix. For instance, the tensor space $\mathbf{R}^{n_1 \times n_2 \times \cdots \times n_m}$ can also be seen as a matrix space $\mathbf{R}^{(n_1 \times n_2 \times \cdots \times n_{m_1}) \times (n_{m_1+1} \times n_{m_1+2} \times \cdots \times n_m)}$, where the row is actually an m_1 array tensor space and the column is another $m - m_1$ array tensor space. Such connections between tensor and matrix re-arrangements will play an important role in this paper. As a convention in this paper, if there is no other specification we shall adhere to the Euclidean norm (i.e., the L_2-norm) for vectors and tensors; in the latter case, the Euclidean norm is also known as the Frobenius norm, and is sometimes denoted as $\|\mathcal{A}\|_F = \sqrt{\sum_{i_1, i_2, \ldots, i_m} \mathcal{A}_{i_1 i_2 \ldots i_m}^2}$. For a given matrix X, we use $\|X\|_*$ to denote the nuclear norm of X, which is the sum of all the singular values of X. Regarding the products, we use $\otimes$ to denote the outer product for tensors; that is, for $\mathcal{A}_1 \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_m}$ and $\mathcal{A}_2 \in \mathbf{R}^{n_{m+1} \times n_{m+2} \times \cdots \times n_{m+\ell}}$, $\mathcal{A}_1 \otimes \mathcal{A}_2$ is in $\mathbf{R}^{n_1 \times n_2 \times \cdots \times n_{m+\ell}}$ with

$$(\mathcal{A}_1 \otimes \mathcal{A}_2)_{i_1 i_2 \ldots i_{m+\ell}} = (\mathcal{A}_1)_{i_1 i_2 \ldots i_m} (\mathcal{A}_2)_{i_{m+1} \ldots i_{m+\ell}}.$$

The inner product between tensors $\mathcal{A}_1$ and $\mathcal{A}_2$ residing in the same space $\mathbf{R}^{n_1 \times n_2 \times \cdots \times n_m}$ is denoted

$$\mathcal{A}_1 \bullet \mathcal{A}_2 = \sum_{i_1, i_2, \ldots, i_m} (\mathcal{A}_1)_{i_1 i_2 \ldots i_m} (\mathcal{A}_2)_{i_1 i_2 \ldots i_m}.$$

Under this light, a multi-linear form $\mathcal{A}(x^1, x^2, \ldots, x^m)$ can also be written in inner/outer products of tensors as

$$\mathcal{A} \bullet (x^1 \otimes \cdots \otimes x^m) := \sum_{i_1, \ldots, i_m} \mathcal{A}_{i_1 i_2 \ldots i_m} (x^1 \otimes \cdots \otimes x^m)_{i_1 \ldots i_m}$$

$$= \sum_{i_1, \ldots, i_m} \mathcal{A}_{i_1 \ldots i_m} \prod_{k=1}^{m} x_{i_k}^k.$$

In the subsequent analysis, for convenience we assume m to be even; i.e., $m = 2d$ in (5), where d is a positive integer. As we will see later, this assumption is essentially non-restrictive. Therefore, we will focus on the following problem of computing the largest eigenvalue of an even order super-symmetric tensor:

$$\begin{aligned} \max \quad & \mathcal{F}(\underbrace{x, \ldots, x}_{2d}) \\ \text{s.t.} \quad & \|x\| = 1, \end{aligned} \tag{6}$$

where $\mathcal{F}$ is a $2d$-th order super-symmetric tensor. In particular, problem (6) can be equivalently written as

$$\max \quad \mathcal{F} \bullet \underbrace{x \otimes \cdots \otimes x}_{2d} \\ \text{s.t.} \quad \|x\| = 1. \tag{7}$$

Given any $2d$-th order super-symmetric tensor form $\mathcal{F}$, we call it *rank one* if $\mathcal{F} = \lambda \underbrace{a \otimes \cdots \otimes a}_{2d}$ for some $a \in \mathbf{R}^n$ and $\lambda \in \{1, -1\}$. Moreover, the CP rank of $\mathcal{F}$ is defined as follows.

Definition 1.1 Suppose $\mathcal{F} \in \mathbf{S}^{n^{2d}}$, the CP rank of $\mathcal{F}$ denoted by rank$(\mathcal{F})$ is the smallest integer r satisfying

$$\mathcal{F} = \sum_{i=1}^{r} \lambda_i \underbrace{a^i \otimes \cdots \otimes a^i}_{2d}, \tag{8}$$

where $a^i \in \mathbf{R}^n, \lambda_i \in \{1, -1\}$.

The idea of decomposing a tensor into an (asymmetric) outer product of vectors was first introduced and studied by Hitchcock [28,29]. This concept of tensor-rank became popular after its rediscovery in the 1970s in the form of CANDECOMP (canonical decomposition) by Carroll and Chang [7] and PARAFAC (parallel factors) by Harshman [24]. Consequently, CANDECOMP and PARAFAC are further abbreviated as 'CP' in the context of 'CP rank' by many authors in the literature.

We remark that, the CP rank is theoretically associated with the complex number field, while in Definition 1.1 decomposition (8) is performed in the real domain. Though the choice of complex or real domain is immaterial for the matrices, it does make a difference in the tensor case [11]. Since we only focus on the real tensors here, throughout this paper we shall use the CP rank to denote the symmetric real rank of a super-symmetric tensor.

In the following, to simplify the notation, we denote $\mathbb{K}(n, d) = \left\{ k = (k_1, \ldots, k_n) \in \mathbb{Z}_+^n \mid \sum_{j=1}^{n} k_j = d \right\}$ and

$$\mathcal{X}_{1^{2k_1} 2^{2k_2} \ldots n^{2k_n}} := \mathcal{X}_{\underbrace{1 \ldots 1}_{2k_1} \underbrace{2 \ldots 2}_{2k_2} \ldots \underbrace{n \ldots n}_{2k_n}}.$$

By letting $\mathcal{X} = \underbrace{x \otimes \cdots \otimes x}_{2d}$ we can further convert problem (7) into:

$$\max \quad \mathcal{F} \bullet \mathcal{X} \\ \text{s.t.} \quad \sum_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^{n} k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \ldots n^{2k_n}} = 1, \tag{9} \\ \mathcal{X} \in \mathbf{S}^{n^{2d}}, \ \text{rank}(\mathcal{X}) = 1,$$

where the first equality constraint is due to the fact that $\sum_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^{n} k_j!} \prod_{j=1}^{n} x_j^{2k_j} = \|x\|^{2d} = 1$.

The difficulty of the above problem lies in the dealing of the rank constraint rank$(\mathcal{X}) = 1$. Not only the rank function itself is difficult to deal with, but also determining the rank of a specific given tensor is already a difficult task, which is NP-hard in general [25]. To give an impression of the difficulty involved in computing tensor ranks, note that there is a particular $9 \times 9 \times 9$ tensor (cf. [34]) whose rank is only known to be in between 18 and 23. One way to deal with the difficulty is to convert the tensor optimization problem (9) into a matrix optimization problem. A typical matricization technique is the so-called mode-n matricization [32]. Roughly speaking, given a tensor $\mathcal{A} \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_m}$, its mode-$n$ matricization denoted by $A(n)$ is to arrange the n-th index of $\mathcal{A}$ to be the row index of the resulting matrix and merge all other indices of $\mathcal{A}$ as the column index of $A(n)$. The precise definition of the mode-n matricization is as follows.

Definition 1.2 For a given tensor $\mathcal{A} \in \mathbf{R}^{n_1 \times n_2 \times \cdots \times n_m}$, the matrix $A(n)$ is the associated mode-n matricization. In particular

$$A(n)_{i_n j} := \mathcal{A}_{i_1 i_2 \ldots i_m}, \ \forall \ 1 \le i_k \le n_k, \ 1 \le k \le m,$$

where

$$j = 1 + \sum_{\substack{k=1 \\ k \neq n}}^{m} (i_k - 1) J_k, \ \text{with} J_k = \prod_{\substack{\ell=1 \\ \ell \neq n}}^{k} n_\ell. \tag{10}$$

The so-called n-rank of $\mathcal{A}$ is defined by the vector $[\text{rank}(A(1)), \text{rank}(A(2)), \ldots, \text{rank}(A(m))]$, where its n-th component corresponds to the column rank of the mode-n matrix $A(n)$. The notion of n-rank has been widely used in the problems of tensor decomposition. Recently, Liu et al. [41] and Gandy et al. [18] considered the low-n-rank tensor recovery problem, which were the first attempts to solve low-rank tensor optimization problems. Along this line, Tomioka et al. [54] analyzed the statistical performance of nuclear norm relaxation of the tensor n-rank minimization problem. Chandrasekaran et al. [8] proposed another interesting idea, in particular they directly applied convex relaxation to the tensor rank and obtained a new norm called tensor nuclear norm, which was numerically intractable. Thus, a further semidefinite representable relaxation was introduced. However, the authors did not provide any numerical results for this relaxation. Therefore, in the following we shall introduce a new scheme to unfold a tensor into a matrix, where we use half of the indices of tensor to form the row index of a matrix and use the other half as the column index. Most importantly, in the next section, we manage to establish some connection between the CP rank of the tensor and the rank of the resulting unfolding matrix.

Definition 1.3 For a given super-symmetric even-order tensor $\mathcal{F} \in \mathbf{S}^{n^{2d}}$, we define its square matricization, denoted by $M(\mathcal{F}) \in \mathbf{R}^{n^d \times n^d}$, as the following:

$$M\,(\mathcal{F})_{k\ell} := \mathcal{F}_{i_1 \ldots i_d i_{d+1} \ldots i_{2d}}, \quad 1 \le i_1, \ldots, i_d, i_{d+1}, \ldots, i_{2d} \le n,$$

where

$$k = \sum_{j=1}^{d} (i_j - 1) n^{d-j} + 1, \quad \text{and} \quad \ell = \sum_{j=d+1}^{2d} (i_j - 1) n^{2d-j} + 1.$$

Similarly we introduce below the vectorization of a tensor.

Definition 1.4 The vectorization, $V\,(\mathcal{F})$, of tensor $\mathcal{F} \in \mathbf{R}^{n^m}$ is defined as

$$V\,(\mathcal{F})_k := \mathcal{F}_{i_1 \ldots i_m},$$

where

$$k = \sum_{j=1}^{m} (i_j - 1) n^{m-j} + 1, \, 1 \le i_1, \ldots, i_m \le n.$$

In the same vein, we can convert a vector or a matrix with appropriate dimensions to a tensor. In other words, the inverse of the operators M and V can be defined in the same manner. In the following, we denote $X = M\,(\mathcal{X})$, and so

$$\mathrm{Tr}\,(X) = \sum_{\ell} X_{\ell\ell} \text{ with } \ell = \sum_{j=1}^{d} (i_j - 1) n^{d-j} + 1.$$

If we assume $\mathcal{X}$ to be of rank one, then

$$\mathrm{Tr}\,(X) = \sum_{i_1, \ldots, i_d} \mathcal{X}_{i_1 \ldots i_d i_1 \ldots i_d} = \sum_{i_1, \ldots, i_d} \mathcal{X}_{i_1^2 \ldots i_d^2}.$$

In the above expression, $(i_1, \ldots, i_d)$ is a subset of $(1, 2, \ldots, n)$. Suppose that j appears k_j times in $(i_1, \ldots, i_d)$ with $j = 1, 2, \ldots, n$ and $\sum_{j=1}^{n} k_j = d$. Then for a fixed outcome $(k_1, k_2, \ldots, k_n)$, the total number of permutations $(i_1, \ldots, i_d)$ that can achieve such combination is

$$\binom{d}{k_1} \binom{d-k_1}{k_2} \binom{d-k_1-k_2}{k_3} \cdots \binom{d-k_1-\cdots-k_{n-1}}{k_n} = \frac{d!}{\prod_{j=1}^{n} k_j!}.$$

Consequently,

$$\mathrm{Tr}\,(X) = \sum_{i_1, \ldots, i_d} \mathcal{X}_{i_1^2 \ldots i_d^2} = \sum_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^{n} k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \ldots n^{2k_n}}. \tag{11}$$

In light of the above discussion, if we further denote $F = M\,(\mathcal{F})$, then the objective in (9) is $\mathcal{F} \bullet \mathcal{X} = \mathrm{Tr}\,(FX)$, while the first constraint $\sum_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^{n} k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \ldots n^{2k_n}} =$

$1 \iff \text{Tr}(X) = 1$. The hard constraint in (9) is $\text{rank}(\mathcal{X}) = 1$. It is straightforward to see that if $\mathcal{X}$ is of rank one, then by letting $\mathcal{X} = \lambda \underbrace{x \otimes \cdots \otimes x}_{2d}$ for some $\lambda \in \{1, -1\}$ and $\mathcal{Y} = \underbrace{x \otimes \cdots \otimes x}_{d}$, we have $M(\mathcal{X}) = \lambda V(\mathcal{Y}) V(\mathcal{Y})^\top$, which is to say that matrix $M(\mathcal{X})$ is of rank one too. In the next section we shall continue to show that the other way around is also true.

2 Equivalence under the rank-one hypothesis

We first present some useful observations below.

Lemma 2.1 *Suppose* $\mathcal{A} \in \mathbf{R}^{n^d}$ *is an n dimensional d-th order tensor and* $\mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}$. *Then we have:*

> (i) $\mathcal{A} \in \mathbf{S}^{n^d}$;
> (ii) *the n-rank of* $\mathcal{A}$ *is* $[1, 1, \ldots, 1]$.

Proof We denote $\mathcal{F} = \mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}$. For any d-tuples $\{i_1, \ldots, i_d\}$, and one of its permutations $\{j_1, \ldots, j_d\} \in \pi(i_1, \ldots, i_d)$, it holds that

$$
\begin{aligned}
\left(\mathcal{A}_{i_1\ldots i_d} - \mathcal{A}_{j_1\ldots j_d}\right)^2 &= \mathcal{A}_{i_1\ldots i_d}^2 + \mathcal{A}_{j_1\ldots j_d}^2 - 2\mathcal{A}_{i_1\ldots i_d}\mathcal{A}_{j_1\ldots j_d} \\
&= \mathcal{F}_{i_1\ldots i_d i_1\ldots i_d} + \mathcal{F}_{j_1\ldots j_d j_1\ldots j_d} - 2\mathcal{F}_{i_1\ldots i_d j_1\ldots j_d} = 0,
\end{aligned}
$$

where the last equality is due to the fact that $\mathcal{F}$ is super-symmetric. Therefore, $\mathcal{A}$ is super-symmetric.

To prove the second statement, we first observe that for any two d-tuples $\{i_1, \ldots, i_d\}$ and $\{i_1', \ldots, i_d'\}$, due to the super-symmetry of $\mathcal{F}$, we have

$$
\mathcal{A}_{i_1\ldots i_d}\mathcal{A}_{i_1'\ldots i_d'} = \mathcal{F}_{i_1\ldots i_d i_1'\ldots i_d'} = \mathcal{F}_{i_1' i_2\ldots i_d i_1 i_2'\ldots i_d'} = \mathcal{A}_{i_1' i_2\ldots i_d}\mathcal{A}_{i_1 i_2'\ldots i_d'}.
$$

Now consider the mode-1 unfolding, which is the matrix $A(1)$. For any two components $A(1)_{i_1 j}$ and $A(1)_{i_1' j'}$ with $j = 1 + \sum_{k=2}^{m}(i_k - 1)J_k$, $j' = 1 + \sum_{k=2}^{m}(i_k' - 1)J_k$ and J_k is defined in (10), the equation above implies that

$$
A(1)_{i_1 j} A(1)_{i_1' j'} = \mathcal{A}_{i_1\ldots i_d}\mathcal{A}_{i_1'\ldots i_d'} = \mathcal{A}_{i_1' i_2\ldots i_d}\mathcal{A}_{i_1 i_2'\ldots i_d'} = A(1)_{i_1' j} A(1)_{i_1 j'}.
$$

Therefore, every 2×2 minor of matrix $A(1)$ is zero and so $A(1)$ is of rank one. Moreover, since $\mathcal{A}$ is super-symmetric, the mode-unfolded matrices are all the same. Thus, we conclude that the n-rank of $\mathcal{A}$ is $[1, 1, \ldots, 1]$. $\square$

The following lemma tells us if a super-symmetric tensor is of rank one in the sense of nonsymmetric CP, then the symmetric CP rank of the tensor is also one.

Lemma 2.2 *If a d-th order tensor $\mathcal{A} = a^1 \otimes a^2 \otimes \cdots \otimes a^d$ is super-symmetric, then we have $a^i = \pm a^1$ for $i = 2, \ldots, d$ and $\mathcal{A} = \lambda \underbrace{a^1 \otimes a^1 \otimes \cdots \otimes a^1}_{d}$ for some $\lambda = \pm 1$.*

Proof Since $\mathcal{A}$ is super-symmetric, from Theorem 4.1 in [10], we know that

$$\max_{\|x\|=1} \left| \mathcal{A} \left(\underbrace{x, \ldots, x}_{d} \right) \right| = \max_{\|x^i\|=1, i=1,\ldots,d} \mathcal{A}(x^1, \ldots, x^d) = \|a^1\| \times \|a^2\| \times \cdots \times \|a^d\|.$$

So there must exist an x^* with $\|x^*\| = 1$ such that $|(a^i)^\top x^*| = \|a^i\|$ for all i, which implies that $a^i = \pm a^1$ for $i = 2, \ldots, d$, and thus the conclusion follows. $\quad\square$

We have the following proposition as the immediate consequence of the above lemmas.

Proposition 2.3 *Suppose $\mathcal{A} \in \mathbf{R}^{n^d}$ is an n dimensional d-th order tensor. The following two statements are equivalent:*

$$(i) \quad \mathcal{A} \in \mathbf{S}^{n^d}, \text{ and } \mathrm{rank}(\mathcal{A}) = 1;$$
$$(ii) \quad \mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}.$$

Proof We shall first show (i) $\Longrightarrow$ (ii). Suppose $\mathcal{A} \in \mathbf{S}^{n^d}$ with $\mathrm{rank}(\mathcal{A}) = 1$. Then there exists a vector $a \in \mathbf{R}^n$ and a scalar $\lambda \in \{1, -1\}$ such that $\mathcal{A} = \lambda \underbrace{a \otimes a \otimes \cdots \otimes a}_{d}$.

Consequently, $\mathcal{A} \otimes \mathcal{A} = \underbrace{a \otimes a \otimes \cdots \otimes a}_{2d} \in \mathbf{S}^{n^{2d}}$.

Now we prove (ii) $\Longrightarrow$ (i). From Lemma 2.1, we know that $\mathcal{A}$ is super-symmetric and the n-rank of $\mathcal{A}$ is $[1, 1, \ldots, 1]$. It is well known that the n-rank of a tensor corresponds to the size of the *core tensor* associated with the *smallest exact Tucker decomposition* [32]. Consequently, the n-rank of $\mathcal{A}$ is $[1, 1, \ldots, 1]$ means that the core tensor associated with the exact Tucker decomposition of $\mathcal{A}$ is a scalar, thus the nonsymmetric real CP rank of $\mathcal{A}$ is also one. Finally, due to Lemma 2.2 and the fact that $\mathcal{A}$ is super-symmetric, we conclude that the symmetric CP rank of $\mathcal{A}$ is one, i.e. $\mathrm{rank}(\mathcal{A}) = 1$. $\quad\square$

Now we are ready to present the main result of this section.

Theorem 2.4 *Suppose $\mathcal{X} \in \mathbf{S}^{n^{2d}}$ and $X = M(\mathcal{X}) \in \mathbf{R}^{n^d \times n^d}$. Then we have*

$$\mathrm{rank}(\mathcal{X}) = 1 \Longleftrightarrow \mathrm{rank}(X) = 1.$$

Proof As remarked earlier, that $\mathrm{rank}(\mathcal{X}) = 1 \Longrightarrow \mathrm{rank}(X) = 1$ is evident. To see this, suppose $\mathrm{rank}(\mathcal{X}) = 1$ and $\mathcal{X} = \underbrace{x \otimes \cdots \otimes x}_{2d}$ for some $x \in \mathbf{R}^n$. By constructing $\mathcal{Y} = \underbrace{x \otimes \cdots \otimes x}_{d}$, we have $X = M(\mathcal{X}) = V(\mathcal{Y}) V(\mathcal{Y})^\top$, which leads to $\mathrm{rank}(X) = 1$.

To prove the other implication, suppose that we have $\mathcal{X} \in \mathbf{S}^{n^{2d}}$ and $\boldsymbol{M}(\mathcal{X})$ is of rank one, i.e., $\boldsymbol{M}(\mathcal{X}) = yy^{\top}$ for some vector $y \in \mathbf{R}^{n^d}$. Then $\mathcal{X} = \boldsymbol{V}^{-1}(y) \otimes \boldsymbol{V}^{-1}(y)$, which combined with Proposition 2.3 implies $\boldsymbol{V}^{-1}(y)$ is supper-symmetric and of rank one. Thus there exists $x \in \mathbf{R}^n$ such that $\boldsymbol{V}^{-1}(y) = \underbrace{x \otimes \cdots \otimes x}_{d}$ and $\mathcal{X} = \underbrace{x \otimes \cdots \otimes x}_{2d}$.

$\square$

3 A nuclear norm penalty approach

According to Theorem 2.4, we know that a super-symmetric tensor is of rank one, if and only if its matrix correspondence obtained via the matricization operation defined in Definition 1.3, is also of rank one. As a result, we can reformulate Problem (9) equivalently as the following matrix optimization problem:

$$
\begin{aligned}
\max \quad & \operatorname{Tr}(FX) \\
\text{s.t.} \quad & \operatorname{Tr}(X) = 1, \ \boldsymbol{M}^{-1}(X) \in \mathbf{S}^{n^{2d}}, \\
& X \in \mathbf{S}^{n^d \times n^d}, \ \operatorname{rank}(X) = 1,
\end{aligned}
\tag{12}
$$

where $X = \boldsymbol{M}(\mathcal{X})$, $F = \boldsymbol{M}(\mathcal{F})$, and $\mathbf{S}^{n^d \times n^d}$ denotes the set of $n^d \times n^d$ symmetric matrices. Note that the constraints $\boldsymbol{M}^{-1}(X) \in \mathbf{S}^{n^{2d}}$ requires the tensor correspondence of X to be super-symmetric, which essentially correspond to $O(n^{2d})$ linear equality constraints. The rank constraint $\operatorname{rank}(X) = 1$ makes the problem intractable. In fact, Problem (12) is NP-hard in general, due to its equivalence to problem (6).

There have been a large amount of work that deal with the low-rank matrix optimization problems. Research in this area was mainly ignited by the recent emergence of compressed sensing [5,12], matrix rank minimization and low-rank matrix completion problems [4,6,51]. The matrix rank minimization seeks a matrix with the lowest rank satisfying some linear constraints, i.e.,

$$
\min_{X \in \mathbf{R}^{n_1 \times n_2}} \operatorname{rank}(X), \ \text{s.t.}, \ \mathcal{C}(X) = b,
\tag{13}
$$

where $b \in \mathbf{R}^p$ and $\mathcal{C} : \mathbf{R}^{n_1 \times n_2} \to \mathbf{R}^p$ is a linear operator. The results in [4,6,51] show that under certain randomness hypothesis on the linear operator $\mathcal{C}$, with high probability the NP-hard problem (13) is equivalent to the following nuclear norm minimization problem, which is a convex programming problem:

$$
\min_{X \in \mathbf{R}^{n_1 \times n_2}} \|X\|_*, \ \text{s.t.}, \ \mathcal{C}(X) = b.
\tag{14}
$$

In other words, the optimal solution to the convex problem (14) is also the optimal solution to the original NP-hard problem (13).

Motivated by the convex nuclear norm relaxation, one way to deal with the rank constraint in (12) is to introduce the nuclear norm term of X, which penalizes high-ranked X's, in the objective function. This yields the following convex optimization formulation:

$$\begin{aligned}
\max \quad & \mathrm{Tr}\,(FX) - \rho\|X\|_* \\
\text{s.t.} \quad & \mathrm{Tr}\,(X) = 1,\ \boldsymbol{M}^{-1}(X) \in \mathbf{S}^{n^{2d}}, \\
& X \in \mathbf{S}^{n^d \times n^d},
\end{aligned} \tag{15}$$

where $\rho > 0$ is a penalty parameter. It is easy to see that if the optimal solution of (15) (denoted by $\tilde{X}$) is of rank one, then $\|\tilde{X}\|_* = \mathrm{Tr}\,(\tilde{X}) = 1$, which is a constant. In this case, the term $-\rho\|X\|_*$ added to the objective function is a constant, which leads to the fact the solution is also optimal with the constraint that X is rank-one. In fact, Problem (15) is the convex relaxation of the following problem

$$\begin{aligned}
\max \quad & \mathrm{Tr}\,(FX) - \rho\|X\|_* \\
\text{s.t.} \quad & \mathrm{Tr}\,(X) = 1,\ \boldsymbol{M}^{-1}(X) \in \mathbf{S}^{n^{2d}}, \\
& X \in \mathbf{S}^{n^d \times n^d},\ \mathrm{rank}(X) = 1,
\end{aligned}$$

which is equivalent to the original problem (12) since $\rho\|X\|_* = \rho\mathrm{Tr}\,(X) = \rho$.

After solving the convex optimization problem (15) and obtaining the optimal solution $\tilde{X}$, if $\mathrm{rank}(\tilde{X}) = 1$, we can find $\tilde{x}$ such that $\boldsymbol{M}^{-1}(\tilde{X}) = \underbrace{\tilde{x} \otimes \cdots \otimes \tilde{x}}_{2d}$, according to Theorem 2.4. In this case, $\tilde{x}$ is the optimal solution to Problem (6). The original tensor PCA problem, or the Z-eigenvalue problem (6), is thus solved to optimality.

Interestingly, we found from our extensive numerical tests that the optimal solution to Problem (15) is a rank-one matrix almost all the time. In the following, we will show this interesting phenomenon by some concrete examples. The first example is taken from [31].

Example 3.1 We consider a super-symmetric tensor $\mathcal{F} \in \mathbf{S}^{3^4}$ defined by

$$\begin{aligned}
& \mathcal{F}_{1111} = 0.2883, \quad \mathcal{F}_{1112} = -0.0031, \quad \mathcal{F}_{1113} = 0.1973, \mathcal{F}_{1122} = -0.2485, \quad \mathcal{F}_{1123} = -0.2939, \\
& \mathcal{F}_{1133} = 0.3847, \quad \mathcal{F}_{1222} = 0.2972, \quad \mathcal{F}_{1223} = 0.1862, \mathcal{F}_{1233} = 0.0919, \quad \mathcal{F}_{1333} = -0.3619, \\
& \mathcal{F}_{2222} = 0.1241, \quad \mathcal{F}_{2223} = -0.3420, \quad \mathcal{F}_{2233} = 0.2127, \mathcal{F}_{2333} = 0.2727, \quad \mathcal{F}_{3333} = -0.3054.
\end{aligned}$$

We want to compute the largest Z-eigenvalue of $\mathcal{F}$.

Since the size of this tensor is small, we used CVX [23] to solve Problem (15) with $F = \boldsymbol{M}(\mathcal{F})$ and $\rho = 10$. It turned out that CVX produced a rank-one solution $\tilde{X} = aa^\top \in \mathbf{R}^{3^2 \times 3^2}$, where

$$\begin{aligned}
a = (& 0.4451, 0.1649, -0.4688, 0.1649, 0.0611, -0.1737, -0.4688, \\
& -0.1737, 0.4938)^\top.
\end{aligned}$$

Thus we get the matrix correspondence of a by reshaping a into a square matrix A:

$$A = [a(1:3), a(4:6), a(7:9)] = \begin{bmatrix} 0.4451 & 0.1649 & -0.4688 \\ 0.1649 & 0.0611 & -0.1737 \\ -0.4688 & -0.1737 & 0.4938 \end{bmatrix}.$$

It is easy to check that A is a rank-one matrix with the nonzero eigenvalue being 1. This further confirms our theory on the rank-one equivalence, i.e., Theorem 2.4. The eigenvector that corresponds to the nonzero eigenvalue of A is given by

$$\tilde{x} = (-0.6671, -0.2472, 0.7027)^\top,$$

which is the optimal solution to Problem (6).

The next example is from a real MRI application studied by Ghosh et al. [19]. In [19], Ghosh et al. studied a fiber detection problem in diffusion MRI, where they tried to extract the geometric characteristics from an antipodally symmetric spherical function (ASSF), which can be described equivalently in the homogeneous polynomial basis constrained to the sphere. They showed that it is possible to extract the maxima and minima of an ASSF by computing the stationary points of a problem in the form of (6) with $d = 2$ and $n = 4$.

Example 3.2 The objective function $\mathcal{F}(x, x, x, x)$ in this example is given by

$$
\begin{aligned}
& 0.74694x_1^4 \; - \; 0.435103x_1^3x_2 \; + \;\; 0.454945x_1^2x_2^2 + 0.0657818x_1x_2^3 + \hspace{3em} x_2^4 \\
+ \;\; & 0.37089x_1^3x_3 - 0.29883x_1^2x_2x_3 - 0.795157x_1x_2^2x_3 + \;\; 0.139751x_2^3x_3 + 1.24733x_1^2x_3^2 \\
+ \; & 0.714359x_1x_2x_3^2 + \;\; 0.316264x_2^2x_3^2 - \;\;\; 0.397391x_1x_3^3 - \;\; 0.405544x_2x_3^3 + 0.794869x_3^4.
\end{aligned}
$$

Again, we used CVX to solve problem (15) with $F = M(\mathcal{F})$ and $\rho = 10$, and a rank-one solution was found with $\tilde{X} = aa^\top$, where

$$a = (0.0001, 0.0116, 0.0004, 0.0116, 0.9984, 0.0382, 0.0004, 0.0382, 0.0015)^\top.$$

By reshaping vector a, we get the following expression of matrix A:

$$
A = [a(1:3), a(4:6), a(7:9)] = \begin{bmatrix} 0.0001 & 0.0116 & 0.0004 \\ 0.0116 & 0.9984 & 0.0382 \\ 0.0004 & 0.0382 & 0.0015 \end{bmatrix}.
$$

It is easy to check that A is a rank-one matrix with 1 being the nonzero eigenvalue. The eigenvector corresponding to the nonzero eigenvalue of A is given by

$$\tilde{x} = (0.0116, 0.9992, 0.0382)^\top,$$

which is also the optimal solution to the original problem (6).

Henceforth we conduct some numerical tests on randomly generated examples. We constructed 4-th order tensor $\mathcal{T}$ with its components drawn randomly from i.i.d. standard normal distribution. The super-symmetric tensor $\mathcal{F}$ in the tensor PCA problem was obtained by symmetrizing $\mathcal{T}$. All the numerical experiments in this paper were conducted on an Intel Core i5-2520M 2.5 GHz computer with 4 GB of RAM, and all the default settings of Matlab 2012b and CVX 1.22 were used for all the tests. We chose $d = 2$ and the dimension of $\mathcal{F}$ in the tensor PCA problem from $n = 3$ to $n = 9$.

	n	rank-1	CPU
Table 1 Frequency of nuclear norm penalty problem (15) having a rank-one solution	3	100	0.21
	4	100	0.56
	5	100	1.31
	6	100	6.16
	7	100	47.84
	8	100	166.61
	9	100	703.82

We chose $\rho = 10$. For each n, we tested 100 random instances. In Table 1, we report the number of instances that produced rank-one solutions. We also report the average CPU time (in seconds) using CVX to solve the problems.

Table 1 shows that for these randomly created tensor PCA problems, the nuclear norm penalty problem (15) *always* gives a rank-one solution, and thus *always* solves the original problem (6) to optimality.

4 Semidefinite programming relaxation

In this section, we study another convex relaxation for Problem (12). Note that the constraint

$$X \in \mathbf{S}^{n^d \times n^d}, \operatorname{rank}(X) = 1$$

in (12) actually implies that X is positive semidefinite. To get a tractable convex problem, we drop the rank constraint and impose a semidefinite constraint to (12) and consider the following SDP relaxation:

$$
\begin{aligned}
(SDR) \max \ & \operatorname{Tr}(FX) \\
\text{s.t.} \ & \operatorname{Tr}(X) = 1, \\
& \boldsymbol{M}^{-1}(X) \in \mathbf{S}^{n^{2d}}, \ X \succeq 0.
\end{aligned}
\tag{16}
$$

Remark that replacing the rank-one constraint by SDP constraint is by now a common and standard practice; see, e.g., [1,21,55]. The next theorem shows that the SDP relaxation (16) is actually closely related to the nuclear norm penalty problem (15).

Theorem 4.1 *Let X^*_{SDR} and $X^*_{PNP}(\rho)$ be the optimal solutions of problems (16) and (15) respectively. Suppose $Eig^+(X)$ and $Eig^-(X)$ are the summations of non-negative eigenvalues and negative eigenvalues of X respectively, i.e.,*

$$Eig^+(X) := \sum_{i:\, \lambda_i(X) \geq 0} \lambda_i(X), \quad Eig^-(X) := \sum_{i:\, \lambda_i(X) < 0} \lambda_i(X).$$

It holds that

$$2(\rho - v) \left| Eig^-(X^*_{PNP}(\rho)) \right| \leq v - F_0,$$

where $F_0 := \max_{1 \le i \le n} \mathcal{F}_{i^{2d}}$ and v is the optimal value of the following optimization problem

$$\begin{aligned} \max \; & \mathrm{Tr}\,(FX) \\ \text{s.t.} \quad & \|X\|_* = 1, \\ & X \in \mathbf{S}^{n^d \times n^d}. \end{aligned} \tag{17}$$

As a result, $\lim_{\rho \to +\infty} \mathrm{Tr}\,(FX^*_{PNP}(\rho)) = \mathrm{Tr}\,(FX^*_{SDR})$.

Proof Observe that $\boldsymbol{M}\,(\underbrace{e^i \otimes \cdots \otimes e^i}_{2d})$, where e^i is the i-th unit vector, is a feasible solution for problem (15) with objective value $\mathcal{F}_{i^{2d}} - \rho$ for all $1 \le i \le n$. Moreover, by denoting $r(\rho) = \left| Eig^-(X^*_{PNP}(\rho)) \right|$, we have

$$\begin{aligned} \|X^*_{PNP}(\rho)\|_* &= Eig^+(X^*_{PNP}(\rho)) + \left| Eig^-(X^*_{PNP}(\rho)) \right| \\ &= \left(Eig^+(X^*_{PNP}(\rho)) + Eig^-(X^*_{PNP}(\rho)) \right) + 2 \left| Eig^-(X^*_{PNP}(\rho)) \right| \\ &= 1 + 2r(\rho). \end{aligned}$$

Since $X^*_{PNP}(\rho)$ is optimal to problem (15), we have

$$\mathrm{Tr}\,(FX^*_{PNP}(\rho)) - \rho\,(1 + 2r(\rho)) \ge \max_{1 \le i \le n} \mathcal{F}_{i^{2d}} - \rho \ge F_0 - \rho. \tag{18}$$

Moreover, since $X^*_{PNP}(\rho)/\|X^*_{PNP}(\rho)\|_*$ is feasible to problem (17), we have

$$\mathrm{Tr}\,(FX^*_{PNP}(\rho)) \le \|X^*_{PNP}(\rho)\|_* \, v = (1 + 2r(\rho))\, v. \tag{19}$$

Combining (19) and (18) yields

$$2(\rho - v)\,r(\rho) \le v - F_0. \tag{20}$$

Notice that $\|X\|_* = 1$ implies $\|X\|_\infty$ is bounded for all feasible $X \in \mathbf{S}^{n^d \times n^d}$, where $\|X\|_\infty$ denotes the largest entry of X in magnitude. Thus the set $\{X^*_{PNP}(\rho) \mid \rho > 0\}$ is bounded. Let X^*_{PNP} be one cluster point of sequence $\{X^*_{PNP}(\rho) \mid \rho > 0\}$. By taking the limit $\rho \to +\infty$ in (20), we have $r(\rho) \to 0$ and thus $X^*_{PNP} \succeq 0$. Consequently, X^*_{PNP} is a feasible solution to problem (16) and $\mathrm{Tr}\,(FX^*_{SDR}) \ge \mathrm{Tr}\,(FX^*_{PNP})$. On the other hand, it is easy to check that for any $0 < \rho_1 < \rho_2$,

$$\mathrm{Tr}\,(FX^*_{SDR}) \le \mathrm{Tr}\,(FX^*_{PNP}(\rho_2)) \le \mathrm{Tr}\,(FX^*_{PNP}(\rho_1)),$$

which implies $\mathrm{Tr}\,(FX^*_{SDR}) \le \mathrm{Tr}\,(FX^*_{PNP})$. Therefore, $\lim_{\rho \to +\infty} \mathrm{Tr}\,(FX^*_{PNP}(\rho)) = \mathrm{Tr}\,(FX^*_{PNP}) = \mathrm{Tr}\,(FX^*_{SDR})$. $\qquad\square$

Theorem 4.1 shows that when ρ goes to infinity in (15), the optimal solution of the nuclear norm penalty problem (15) converges to the optimal solution of the SDP relaxation (16). As we have shown in Table 1 that the nuclear norm penalty problem (15) returns rank-one solutions for all the randomly created tensor PCA problems that we

Table 2 Frequency of SDP relaxation (16) having a rank-one solution

n	rank-1	CPU
3	100	0.14
4	100	0.25
5	100	0.55
6	100	1.16
7	100	2.37
8	100	4.82
9	100	8.89

tested, it is expected that the SDP relaxation (16) will also be likely to give rank-one solutions. In fact, this is indeed the case as shown through the numerical results in Table 2. As in Table 1, we tested 100 random instances for each n. In Table 2, we report the number of instances that produced rank-one solutions for $d = 2$. We also report the average CPU time (in seconds) using CVX to solve the problems. As we see from Table 2, for these randomly created tensor PCA problems, the SDP relaxation (16) *always* gives a rank-one solution, and thus *always* solves the original problem (6) to optimality.

5 Alternating direction method of multipliers

The computational times reported in Tables 1 and 2 suggest that it can be time-consuming to solve the convex problems (15) and (16) when the problem size is large [especially for the nuclear norm penalty problem (15)]. In this section, we propose an alternating direction method of multipliers (ADMM) for solving (15) and (16) that fully takes advantage of the structures. ADMM is closely related to some operator-splitting methods, known as Douglas–Rachford and Peaceman–Rachford methods, that were proposed in 1950s for solving variational problems arising from PDEs (see [13,47]). These operator-splitting methods were extensively studied later in the literature for finding the zeros of the sum of monotone operators and for solving convex optimization problems (see [14–16,20,40]). The ADMM we will study in this section was shown to be equivalent to the Douglas–Rachford operator-splitting method applied to convex optimization problem (see [17]). ADMM was revisited recently as it was found to be very efficient for many sparse and low-rank optimization problems arising from the recent emergence of compressed sensing [59], compressive imaging [22,57], robust PCA [53], sparse inverse covariance selection [52,60], sparse PCA [42] and SDP [58] etc. For a more complete discussion and list of references on ADMM, we refer to the recent survey paper by Boyd et al. [3] and the references therein.

Generally speaking, ADMM solves the following convex optimization problem,

$$
\begin{aligned}
\min_{x\in\mathbf{R}^n, y\in\mathbf{R}^p} \quad & f(x) + g(y) \\
\text{s.t.} \quad & Ax + By = b \\
& x \in \mathcal{C}, \ y \in \mathcal{D},
\end{aligned}
\tag{21}
$$

where f and g are convex functions, $A \in \mathbf{R}^{m \times n}$, $B \in \mathbf{R}^{m \times p}$, $b \in \mathbf{R}^m$, $\mathcal{C}$ and $\mathcal{D}$ are some simple convex sets. A typical iteration of ADMM for solving (21) can be described as follows:

$$\begin{cases} x^{k+1} := \mathrm{argmin}_{x \in \mathcal{C}} \ \mathcal{L}_\mu(x, y^k; \lambda^k) \\ y^{k+1} := \mathrm{argmin}_{y \in \mathcal{D}} \ \mathcal{L}_\mu(x^{k+1}, y; \lambda^k) \\ \lambda^{k+1} := \lambda^k - (Ax^{k+1} + By^{k+1} - b)/\mu, \end{cases} \qquad (22)$$

where the augmented Lagrangian function $\mathcal{L}_\mu(x, y; \lambda)$ is defined as

$$\mathcal{L}_\mu(x, y; \lambda) := f(x) + g(y) - \langle \lambda, Ax + By - b \rangle + \frac{1}{2\mu} \|Ax + By - b\|^2,$$

with λ being the Lagrange multiplier and $\mu > 0$ a penalty parameter. The following theorem gives the global convergence of (22) for solving (21), and this has been well studied in the literature (see, e.g., [14,16]).

Theorem 5.1 *Assume both A and B are of full column rank, the sequence $\{(x^k, y^k, \lambda^k)\}$ generated by (22) globally converges to a pair of primal and dual optimal solutions (x^*, y^*) and λ^* of (21) from any starting point.*

Because both the nuclear norm penalty problem (15) and SDP relaxation (16) can be rewritten in the form of (21), we can apply ADMM to solve them.

5.1 ADMM for nuclear norm penalty problem (15)

Note that the nuclear norm penalty problem (15) can be rewritten equivalently as

$$\begin{aligned} \min \ & -\mathrm{Tr}\,(FY) + \rho \|Y\|_* \\ \mathrm{s.t.} \ & X - Y = 0, \\ & X \in \mathcal{C}, \end{aligned} \qquad (23)$$

where $\mathcal{C} := \{X \in \mathbf{S}^{n^d \times n^d} \mid \mathrm{Tr}\,(X) = 1, \ \boldsymbol{M}^{-1}(X) \in \mathbf{S}^{n^{2d}}\}$. A typical iteration of ADMM for solving (23) can be described as

$$\begin{cases} X^{k+1} := \mathrm{argmin}_{X \in \mathcal{C}} -\mathrm{Tr}\,(FY^k) + \rho \|Y^k\|_* - \langle \Lambda^k, X - Y^k \rangle + \frac{1}{2\mu} \|X - Y^k\|_F^2 \\ Y^{k+1} := \mathrm{argmin}_Y -\mathrm{Tr}\,(FY) + \rho \|Y\|_* - \langle \Lambda^k, X^{k+1} - Y \rangle + \frac{1}{2\mu} \|X^{k+1} - Y\|_F^2 \\ \Lambda^{k+1} := \Lambda^k - (X^{k+1} - Y^{k+1})/\mu, \end{cases}$$

$$(24)$$

where Λ is the Lagrange multiplier associated with the equality constraint in (23) and $\mu > 0$ is a penalty parameter. Following Theorem 5.1, we know that the sequence $\{(X^k, Y^k, \Lambda^k)\}$ generated by (24) globally converges to a pair of primal and dual optimal solutions (X^*, Y^*) and Λ^* of (23) from any starting point.

Next we show that the two subproblems in (24) are both easy to solve. The first subproblem in (24) can be equivalently written as

$$X^{k+1} := \mathrm{argmin}_{X \in \mathcal{C}} \frac{1}{2} \|X - (Y^k + \mu \Lambda^k)\|_F^2, \qquad (25)$$

i.e., the solution of the first subproblem in (24) corresponds to the projection of $Y^k + \mu\Lambda^k$ onto convex set $\mathcal{C}$. We will elaborate how to compute this projection in Sect. 5.2.

The second subproblem in (24) can be reduced to:

$$Y^{k+1} := \underset{Y}{\operatorname{argmin}} \quad \mu\rho\|Y\|_* + \frac{1}{2}\|Y - (X^{k+1} - \mu(\Lambda^k - F))\|_F^2. \qquad (26)$$

This problem is known to have a closed-form solution that is given by the following so-called matrix shrinkage operation (see, e.g., [43]):

$$Y^{k+1} := U\operatorname{Diag}\left(\max\left\{\sigma - \mu\rho, 0\right\}\right)V^\top,$$

where $U\operatorname{Diag}(\sigma)V^\top$ is the singular value decomposition of matrix $X^{k+1} - \mu(\Lambda^k - F)$.

5.2 The projection

In this subsection, we study how to solve (25), i.e., how to compute the following projection for any given matrix $Z \in \mathbf{S}^{n^d \times n^d}$:

$$\begin{aligned} \min \quad & \|X - Z\|_F^2 \\ \text{s.t.} \quad & \operatorname{Tr}(X) = 1, \\ & M^{-1}(X) \in \mathbf{S}^{n^{2d}}. \end{aligned} \qquad (27)$$

For the sake of discussion, in the following we consider the equivalent tensor representation of (27):

$$\begin{aligned} \min \quad & \|\mathcal{X} - \mathcal{Z}\|_F^2 \\ \text{s.t.} \quad & \sum_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^n k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \ldots n^{2k_n}} = 1, \\ & \mathcal{X} \in \mathbf{S}^{n^{2d}}, \end{aligned} \qquad (28)$$

where $\mathcal{X} = M^{-1}(X)$, $\mathcal{Z} = M^{-1}(Z)$, and the equality constraint is due to (11). Now we denote index set

$$\mathbf{I} = \left\{ (i_1 \ldots i_{2d}) \in \pi(1^{2k_1} \ldots n^{2k_n}) \mid k = (k_1, \ldots, k_n) \in \mathbb{K}(n,d) \right\}.$$

Then the first-order optimality conditions of Problem (28) imply

$$\begin{cases} 2\left(|\pi(i_1 \ldots i_{2d})|\, \mathcal{X}_{i_1 \ldots i_{2d}} - \displaystyle\sum_{j_1 \ldots j_{2d} \in \pi(i_1 \ldots i_{2d})} \mathcal{Z}_{j_1 \ldots j_{2d}}\right) = 0, & \text{if } (i_1 \ldots i_{2d}) \notin \mathbf{I}, \\[2ex] 2\left(\dfrac{(2d)!}{\prod_{j=1}^n (2k_j)!}\, \mathcal{X}_{1^{2k_1} \ldots n^{2k_n}} - \displaystyle\sum_{j_1 \ldots j_{2d} \in \pi(1^{2k_1} \ldots n^{2k_n})} \mathcal{Z}_{j_1 \ldots j_{2d}}\right) - \lambda \dfrac{(d)!}{\prod_{j=1}^n (k_j)!} = 0, & \text{otherwise.} \end{cases}$$

Denote $\hat{\mathcal{Z}}$ to be the super-symmetric counterpart of tensor $\mathcal{Z}$, i.e.,

$$\hat{\mathcal{Z}}_{i_1\dots i_{2d}} = \sum_{j_1\dots j_{2d}\in\pi(i_1\dots i_{2d})} \frac{\mathcal{Z}_{j_1\dots j_{2d}}}{|\pi(i_1\dots i_{2d})|}$$

and $\alpha(k,d) := \left(\frac{(d)!}{\prod_{j=1}^{n}(k_j)!}\right) / \left(\frac{(2d)!}{\prod_{j=1}^{n}(2k_j)!}\right)$. Then due to the first-order optimality conditions for (28), the optimal solution $\mathcal{X}^*$ of Problem (28) satisfies

$$\begin{cases} \mathcal{X}^*_{i_1\dots i_{2d}} = \hat{\mathcal{Z}}_{i_1\dots i_{2d}}, & \text{if } (i_1\dots i_{2d}) \notin \mathbf{I}, \\ \mathcal{X}^*_{1^{2k_1}\dots n^{2k_n}} = \frac{\lambda}{2}\alpha(k,d) + \hat{\mathcal{Z}}_{1^{2k_1}\dots n^{2k_n}}, & \text{otherwise}. \end{cases} \tag{29}$$

Multiplying the second equality of (29) by $\frac{(d)!}{\prod_{j=1}^{n}(k_j)!}$ and summing the resulting equality over all $k = (k_1,\dots,k_n)$ yield

$$\sum_{k\in\mathbb{K}(n,d)} \frac{(d)!}{\prod_{j=1}^{n}(k_j)!}\mathcal{X}^*_{1^{2k_1}\dots n^{2k_n}} = \frac{\lambda}{2}\sum_{k\in\mathbb{K}(n,d)} \frac{(d)!}{\prod_{j=1}^{n}(k_j)!}\alpha(k,d)$$

$$+ \sum_{k\in\mathbb{K}(n,d)} \frac{(d)!}{\prod_{j=1}^{n}(k_j)!}\hat{\mathcal{Z}}_{1^{2k_1}\dots n^{2k_n}}.$$

It remains to determine λ. Noticing that $\mathcal{X}^*$ is a feasible solution for problem (28), we have $\sum_{k\in\mathbb{K}(n,d)} \frac{(d)!}{\prod_{j=1}^{n}(k_j)!}\mathcal{X}^*_{1^{2k_1}\dots n^{2k_n}} = 1$. As a result,

$$\lambda = 2\left(1 - \sum_{k\in\mathbb{K}(n,d)} \frac{(d)!}{\prod_{j=1}^{n}(k_j)!}\hat{\mathcal{Z}}_{1^{2k_1}\dots n^{2k_n}}\right) / \sum_{k\in\mathbb{K}(n,d)} \frac{(d)!}{\prod_{j=1}^{n}(k_j)!}\alpha(k,d),$$

and thus we derived $\mathcal{X}^*$ and $X^* = \boldsymbol{M}(\mathcal{X}^*)$ as the desired optimal solution for (27).

5.3 ADMM for SDP relaxation (16)

Note that the SDP relaxation problem (16) can be formulated as

$$\begin{aligned} \min \quad & -\text{Tr}(FY) \\ \text{s.t.} \quad & \text{Tr}(X) = 1, \quad \boldsymbol{M}^{-1}(X) \in \mathbf{S}^{n^{2d}} \\ & X - Y = 0, \quad Y \succeq 0. \end{aligned} \tag{30}$$

A typical iteration of ADMM for solving (30) is

$$\begin{cases} X^{k+1} := \text{argmin}_{X\in\mathcal{C}} -\text{Tr}(FY^k) - \langle \Lambda^k, X - Y^k \rangle + \frac{1}{2\mu}\|X - Y^k\|_F^2 \\ Y^{k+1} := \text{argmin}_{Y\succeq 0} -\text{Tr}(FY) - \langle \Lambda^k, X^{k+1} - Y \rangle + \frac{1}{2\mu}\|X^{k+1} - Y\|_F^2 \\ \Lambda^{k+1} := \Lambda^k - (X^{k+1} - Y^{k+1})/\mu, \end{cases} \tag{31}$$

where $\mu > 0$ is a penalty parameter. Following Theorem 5.1, we know that the sequence $\{(X^k, Y^k, \Lambda^k)\}$ generated by (31) globally converges to a pair of primal and dual optimal solutions (X^*, Y^*) and Λ^* of (30) from any starting point.

It is easy to check that the two subproblems in (31) are both relatively easy to solve. Specifically, the solution of the first subproblem in (31) corresponds to the projection of $Y^k + \mu\Lambda^k$ onto $\mathcal{C}$. The solution of the second problem in (31) corresponds to the projection of $X^{k+1} + \mu F - \mu\Lambda^k$ onto the positive semidefinite cone $Y \succeq 0$, i.e.,

$$Y^{k+1} := U \text{Diag} \left(\max\{\sigma, 0\} \right) U^\top,$$

where $U \text{Diag} (\sigma) U^\top$ is the eigenvalue decomposition of matrix $X^{k+1} + \mu F - \mu\Lambda^k$.

6 Numerical results

6.1 The ADMM for convex programs (15) and (16)

In this subsection, we report the results on using ADMM (24) to solve the nuclear norm penalty problem (15) and ADMM (31) to solve the SDP relaxation (16). For the nuclear norm penalty problem (15), we choose $\rho = 10$. For ADMM, we choose $\mu = 0.5$ and we terminate the algorithms whenever

$$\frac{\|X^k - X^{k-1}\|_F}{\|X^{k-1}\|_F} + \|X^k - Y^k\|_F \leq 10^{-6}.$$

We shall compare ADMM and CVX for solving (15) and (16), using the default solver of CVX—SeDuMi version 1.21. We report in Table 3 the results on randomly created problems with $d = 2$ and $n = 6, 7, 8, 9$. For each pair of d and n, we test ten randomly created examples. In Table 3, we use 'Inst.' to denote the number of the instance and use 'Iter.' to denote the number of iterations for ADMM to solve a random instance. We use 'Sol.Dif.' to denote the relative difference of the solutions obtained by ADMM and CVX, i.e., Sol.Dif. $= \frac{\|X_{ADMM} - X_{CVX}\|_F}{\max\{1, \|X_{CVX}\|_F\}}$, and we use 'Val.Dif.' to denote the relative difference of the objective values obtained by ADMM and CVX, i.e., Val.Dif. $= \frac{|v_{ADMM} - v_{CVX}|}{\max\{1, |v_{CVX}|\}}$. We use T_{ADMM} and T_{CVX} to denote the CPU times (in seconds) of ADMM and CVX, respectively. From Table 3 we see that, ADMM produced comparable solutions compared to CVX; however, ADMM were much faster than CVX, i.e., the interior point solver, especially for nuclear norm penalty problem (15). Note that when $n = 10$, ADMM was about 500 times faster than CVX for solving (15), and was about 8 times faster for solving (16).

In Table 4, we report the results on larger problems, i.e., $n = 14, 16, 18, 20$. Because it becomes time consuming to use CVX to solve the nuclear norm penalty problem (15) (our numerical tests showed that it took more than three hours to solve one instance of (15) for $n = 11$ using CVX), we compare the solution quality and objective value of the solution generated by ADMM for solving (15) with solution generated by CVX for solving SDP problem (16). From Table 4 we see that, the

Table 3 Comparison of CVX and ADMM for small-scale problems

Inst. #	Nuclear norm penalty (15)					SDP (16)				
	Sol.Dif.	Val.Dif.	T_{ADMM}	Iter.	T_{CVX}	Sol.Dif.	Val.Dif.	T_{ADMM}	Iter.	T_{CVX}
Dimension n = 6										
1	1.77e−04	3.28e−06	1.16	464	18.50	1.01e−04	2.83e−06	0.50	367	1.98
2	1.25e−04	3.94e−07	0.71	453	13.43	4.99e−05	3.78e−06	0.38	355	1.68
3	1.56e−04	2.36e−07	0.89	478	12.20	4.59e−05	3.51e−06	0.39	370	1.33
4	3.90e−05	6.91e−07	0.59	475	14.10	8.00e−05	9.57e−07	0.44	364	2.63
5	1.49e−04	3.69e−06	0.58	459	15.08	4.74e−05	3.18e−06	0.60	355	1.98
6	8.46e−05	3.92e−06	1.07	463	13.23	1.02e−04	2.68e−07	0.76	362	1.46
7	5.59e−05	4.12e−06	0.86	465	12.62	4.91e−05	4.75e−06	0.37	344	1.54
8	5.24e−05	3.95e−06	0.61	462	14.07	1.63e−05	2.97e−06	0.55	368	1.90
9	9.30e−05	3.05e−06	0.85	471	11.41	1.05e−04	2.90e−06	0.39	380	1.39
10	1.36e−04	3.89e−08	0.56	465	11.04	3.38e−05	3.11e−06	0.30	319	1.69
Dimension n = 7										
1	1.59e−04	4.62e−07	1.23	600	65.73	1.14e−04	4.09e−06	0.82	453	2.60
2	9.11e−05	3.93e−07	1.02	593	68.65	8.24e−05	2.87e−09	0.79	474	2.51
3	2.61e−04	4.19e−06	1.07	609	66.08	6.83e−05	4.01e−06	0.78	480	2.53
4	1.12e−04	4.44e−06	1.07	590	65.21	6.02e−05	3.88e−06	0.86	480	2.50
5	1.22e−04	4.34e−06	1.10	614	57.40	9.15e−05	4.15e−07	0.81	487	2.57
6	1.44e−04	8.81e−08	1.06	599	60.89	4.51e−05	4.46e−06	0.77	466	2.44
7	1.93e−04	3.81e−06	1.08	590	66.09	1.19e−04	2.82e−07	0.62	389	2.54
8	1.53e−04	4.59e−06	1.09	594	59.98	2.76e−05	3.73e−06	0.75	463	2.61
9	1.41e−04	4.29e−08	1.06	616	78.20	3.29e−04	4.21e−06	0.69	443	2.57
10	1.51e−04	3.94e−06	0.83	501	75.58	1.23e−04	3.52e−06	0.78	454	2.63
Dimension n = 8										
1	2.86e−04	5.10e−06	2.15	728	342.25	1.12e−04	4.52e−06	1.59	592	5.34
2	2.76e−04	3.95e−07	2.07	739	303.75	8.17e−05	4.78e−06	1.81	591	5.02
3	9.29e−05	4.78e−06	7.74	2,864	333.46	2.57e−05	5.00e−06	7.20	2,746	4.75
4	3.21e−04	4.65e−06	2.01	715	337.57	9.86e−05	4.01e−06	1.47	512	5.00
5	1.26e−04	7.05e−07	1.92	746	335.63	7.41e−05	4.36e−06	1.68	607	4.92
6	1.32e−04	1.63e−07	2.12	745	336.35	7.80e−05	5.00e−06	1.44	550	5.29
7	3.49e−04	7.19e−07	2.00	739	309.76	6.33e−05	4.55e−07	1.54	582	5.03
8	4.55e−05	4.72e−07	2.13	744	316.74	3.59e−05	7.27e−07	1.59	600	5.02
9	5.60e−04	4.99e−06	2.06	759	336.10	4.19e−05	4.97e−06	1.46	569	6.00
10	2.65e−04	1.36e−07	2.46	746	382.20	8.00e−05	4.14e−06	1.86	606	5.98
Dimension n = 9										
1	1.41e−04	1.35e−07	4.35	910	1,370.60	7.29e−05	4.78e−06	3.26	715	12.61
2	1.83e−04	5.77e−06	3.60	872	1,405.46	1.77e−04	4.72e−06	2.86	732	9.63
3	4.00e−04	4.85e−06	3.24	807	1,709.30	3.12e−04	8.28e−07	2.73	702	9.99
4	3.34e−04	1.36e−07	3.06	747	1,445.57	6.13e−05	3.19e−07	2.91	707	10.19
5	2.63e−04	5.43e−06	3.62	904	1,307.60	2.34e−05	4.68e−06	2.82	729	10.20
6	8.01e−05	9.01e−08	3.78	906	1,353.45	9.33e−05	5.37e−06	2.49	597	9.31
7	2.30e−04	5.16e−06	3.77	900	1,434.71	8.14e−05	5.68e−06	2.75	676	9.52
8	3.27e−04	5.45e−06	3.71	908	1,314.14	1.98e−05	5.10e−06	2.91	730	9.98
9	9.53e−05	5.56e−06	3.66	888	1,575.16	1.69e−04	4.82e−06	2.85	714	9.64
10	2.73e−04	2.16e−07	4.50	1,136	1,628.80	2.73e−05	4.98e−06	3.39	882	9.90

Table 4 Comparison of CVX and ADMM for large-scale problems

Inst. #	NNP				SDP				
	Sol.Dif.$_{DS}$	Val.Dif.$_{DS}$	T_{ADMM}	Iter.	Sol.Dif.	Val.Dif.	T_{ADMM}	Iter.	T_{CVX}
Dimension n = 14									
1	4.61e−04	8.41e−06	36.85	1,913	4.61e−04	8.35e−06	37.00	1,621	158.21
2	4.02e−04	2.94e−07	39.52	1,897	4.02e−04	7.93e−06	39.65	1,639	167.89
3	1.62e−04	2.68e−08	37.21	1,880	1.62e−04	8.23e−06	34.36	1,408	213.04
4	4.92e−04	7.74e−06	45.15	1,918	4.92e−04	4.70e−07	59.84	1,662	202.95
5	8.56e−04	8.15e−06	34.93	1,674	8.56e−04	8.14e−06	38.15	1,588	194.01
6	3.99e−05	4.05e−07	34.41	1,852	4.08e−05	7.48e−06	32.28	1,411	186.99
7	7.98e−05	7.90e−06	38.11	1,839	7.94e−05	3.76e−08	40.81	1,555	191.76
8	1.50e−04	8.10e−06	38.29	1,990	1.50e−04	8.30e−06	34.10	1,543	164.13
9	1.35e−04	8.54e−06	34.58	1,874	1.35e−04	2.62e−07	30.33	1,387	171.77
10	5.50e−04	8.59e−06	37.28	1,825	5.50e−04	7.71e−06	35.85	1,567	169.51
Dimension n = 16									
1	5.22e−05	9.00e−06	125.24	2,359	5.21e−05	9.45e−06	102.85	2,035	582.19
2	1.02e−04	3.37e−07	92.37	2,244	1.02e−04	9.11e−06	63.02	1,427	606.70
3	2.02e−05	5.97e−07	96.21	2,474	2.01e−05	4.40e−07	83.92	1,910	566.92
4	8.53e−05	9.27e−06	90.83	2,323	8.54e−05	9.59e−06	93.44	2,048	560.54
5	2.14e−04	9.19e−06	86.22	2,359	2.14e−04	2.19e−07	80.06	1,961	523.15
6	3.12e−04	9.29e−06	88.82	2,304	3.12e−04	8.58e−06	88.31	2,042	498.55
7	9.69e−05	9.12e−06	88.29	2,431	9.65e−05	2.86e−07	88.05	2,067	520.82
8	3.34e−04	1.00e−05	85.32	2,271	3.34e−04	8.53e−06	85.04	2,043	515.85
9	2.61e−04	9.01e−06	93.13	2,475	2.61e−04	9.12e−06	88.85	2,034	505.71
10	2.06e−04	3.45e−07	103.92	2,813	2.05e−04	1.01e−05	94.41	2,269	527.50
Dimension n = 18									
1	2.70e−04	1.01e−05	172.97	2,733	2.70e−04	1.87e−07	168.91	2,323	1,737.94
2	8.17e−04	1.11e−05	184.70	2,970	8.17e−04	1.99e−07	168.83	2,365	1,549.10
3	1.07e−04	3.22e−08	183.72	2,920	1.07e−04	1.14e−05	169.64	2,456	1,640.04
4	5.16e−04	1.01e−05	182.40	2,958	5.16e−04	1.02e−05	174.72	2,442	1,636.86
5	9.48e−04	1.03e−05	184.69	3,039	9.48e−04	1.04e−05	170.68	2,441	1,543.41
6	1.67e−04	1.03e−05	171.71	2,845	1.67e−04	9.96e−06	182.37	2,553	1,633.55
7	4.87e−05	3.77e−07	180.64	2,883	4.87e−05	2.79e−07	187.56	2,545	1,638.38
8	8.28e−05	1.07e−05	178.35	2,904	8.28e−05	1.04e−05	181.57	2,542	1,641.56
9	2.45e−04	1.06e−07	174.82	2,902	2.45e−04	9.97e−06	152.58	2,127	1,735.26
10	9.58e−05	7.61e−07	191.06	2,872	9.66e−05	1.11e−05	183.29	2,480	1,642.33
Dimension n = 20									
1	1.23e−03	6.98e−08	414.62	3,415	1.23e−03	4.21e−08	388.36	2,810	6,116.02
2	7.93e−04	1.24e−05	401.54	3,383	7.93e−04	1.14e−05	347.27	2,689	6,182.56
3	3.11e−04	1.21e−05	426.91	3,498	3.11e−04	1.21e−05	399.92	2,845	6,808.99
4	7.16e−05	6.99e−07	397.69	3,312	7.40e−05	1.18e−05	366.82	2,758	7,701.91
5	6.24e−04	1.19e−05	435.05	3,564	6.25e−04	1.20e−05	419.23	2,903	7,419.43
6	1.09e−04	1.20e−05	393.25	3,376	1.09e−04	1.15e−05	397.43	2,869	8,622.19
7	4.58e−04	3.21e−05	429.38	3,536	4.58e−04	3.20e−05	422.72	2,938	9,211.37
8	6.15e−04	1.11e−05	273.33	2,330	6.15e−04	7.14e−07	205.49	1,511	5,166.66
9	4.92e−04	1.16e−05	344.99	3,017	4.92e−04	2.32e−07	259.18	1,896	5,063.00
10	3.45e−004	2.56e−004	395.63	3,357	1.14e−005	4.36e−007	359.13	2,713	6,559.39

nuclear norm penalty problem (15) and the SDP problem (16) indeed produce the same solution as they are both close enough to the solution produced by CVX. We also see that using ADMM to solve (15) and (16) were much faster than using CVX to solve (16).

6.2 Comparison with SOS and MBI

Based on the results of the above tests, we may conclude that it is mostly efficient to solve the SDP relaxation by ADMM. In this subsection, we compare this approach with two competing methods: one is based on the Sum of Squares (SOS) approach (Lasserre [35,36] and Parrilo [45,46]), and the other one is the Maximum Block Improvement (MBI) method proposed by Chen et al. [10].

Theoretically speaking, the SOS can solve any general polynomial problems to any given accuracy, but it requires to solve a sequence of (possibly large) semidefinite programs, which limits the applicability of the method to solve large size problems. Henrion et al. [26] developed a specialized Matlab toolbox known as GloptiPoly 3 based on SOS approach, which will be used in our tests. The MBI is tailored for multi-block optimization problem, and the polynomial optimization can be treated as multi-block problems, to which MBI can be applied. As we mentioned before, MBI aims to finding a stationary point, which may or may not be globally optimal.

In Table 5 we report the results using ADMM to solve SDP relaxation of PCA problem and compare them with the results of applying the SOS method as well as the MBI method for the same problem. When using the MBI, as suggested in [10], we actually work on an equivalent problem of (6): $\max_{\|x\|=1} \mathcal{F}(\underbrace{x, \ldots, x}_{2d}) + 6(x^\top x)^d$,

where the equivalence is due to the constraint $\|x\| = 1$. This transformation can help the MBI avoid getting trapped in a local minimum.

We use 'Val.' to denote the objective value of the solution, 'Status' to denote optimal status of GloptiPoly 3, i.e., Status $= 1$ means GloptiPoly 3 successfully identified the optimality of current solution, 'Sol.R.' to denote the solution rank returned by SDP relaxation and thanks to the previous discussion 'Sol.R. $= 1$' means the current solution is already optimal. From Table 5, we see that the MBI is the fastest among all the methods but usually cannot guarantee global optimality, while GloptiPoly 3 is very time consuming but can globally solve most instances. Note that when $n = 20$, our ADMM was about 30 times faster than GloptiPoly 3. Moreover, for some instances GloptiPoly 3 cannot identify the optimality even though the current solution is actually already optimal (see instance 9 with $n = 16$ and instance 3 with $n = 18$).

6.3 Comparison with Z-eigenvalue methods

Qi et al. [49] proposed two heuristic methods to find the maximum Z-eigenvalue of the third order super-symmetric tensors. We will show later in Sect. 8.2 that our method can solve tri-linear (not necessary super-symmetric) tensor PCA problems. Thus in this subsection, we report the results of using ADMM to solve SDP relaxation of the third order tensor PCA problems and compare them with the results of applying the

Table 5 Comparison of SDP relaxation by ADMM with GloptiPoly 3 and MBI

Inst. #	MBI		GLP			SDP by ADMM		
	Val.	Time	Val.	Time	Status	Val.	Time	Sol.R.
Dimension n = 14								
1	5.17	0.23	5.28	143.14	1	5.28	14.29	1
2	5.04	0.22	5.65	109.65	1	5.65	32.64	1
3	5.08	0.13	5.80	119.48	1	5.80	34.30	1
4	5.94	0.16	5.95	100.39	1	5.95	30.64	1
5	4.74	0.48	5.88	122.19	1	5.88	33.13	1
6	5.68	0.54	6.38	122.44	1	6.38	33.30	1
7	4.61	0.12	5.91	104.68	1	5.91	30.17	1
8	5.68	0.23	6.31	141.52	1	6.31	41.73	1
9	5.93	0.22	6.40	102.73	1	6.40	37.32	1
10	5.09	0.36	6.03	114.35	1	6.03	35.68	1
Dimension n = 16								
1	6.52	0.45	6.74	420.10	1	6.74	91.80	1
2	5.51	1.21	5.93	428.10	1	5.93	83.90	1
3	5.02	0.30	6.44	393.16	1	6.44	90.16	1
4	5.60	0.32	6.48	424.07	1	6.48	90.67	1
5	5.78	0.36	6.53	431.44	1	6.53	95.48	1
6	5.23	0.26	6.42	437.58	1	6.42	98.19	1
7	6.11	0.24	6.23	406.16	1	6.23	89.21	1
8	5.92	0.51	6.39	416.58	1	6.39	89.75	1
9	5.47	0.28	6.00	457.29	0	6.00	77.56	1
10	4.95	0.35	6.32	367.26	1	6.32	80.38	1
Dimension n = 18								
1	6.16	0.57	7.38	1,558.00	1	7.38	199.44	1
2	5.94	0.25	6.65	1,388.45	1	6.65	190.52	1
3	7.42	0.22	7.42	1,500.05	0	7.42	193.27	1
4	5.85	0.94	7.21	1,481.34	1	7.21	195.02	1
5	7.35	0.43	7.35	1,596.00	1	7.35	117.44	1
6	5.91	1.05	6.79	1,300.82	1	6.78	193.36	1
7	5.80	0.85	6.84	1,433.50	1	6.84	182.58	1
8	5.72	0.54	6.96	1,648.63	1	6.96	231.88	1
9	6.15	0.17	7.07	1,453.82	1	7.07	212.50	1
10	6.01	1.11	6.89	1,432.06	1	6.89	199.26	1
Dimension n = 20								
1	5.95	0.39	7.40	8,981.97	1	7.40	429.64	1
2	6.13	2.14	6.93	9,339.06	1	6.93	355.25	1
3	6.37	2.49	6.68	9,629.04	1	6.68	418.11	1
4	6.23	1.14	6.87	10,148.21	1	6.87	404.18	1
5	6.62	1.66	7.72	11,079.94	1	7.72	326.44	1
6	6.81	1.26	7.46	10,609.65	1	7.46	415.69	1
7	7.80	1.02	7.80	9,723.37	1	7.80	430.76	1
8	6.03	0.95	7.02	12,755.35	1	7.02	416.00	1
9	7.80	0.61	7.80	12,353.47	1	7.80	430.45	1
10	7.47	0.89	7.47	11,629.12	1	7.47	375.52	1

Table 6 Comparison with Z-eigenvalue methods with data generated by normal distribution

n	SDP	Z1	Z2
4	999	979	995
5	999	964	993
6	1,000	947	998
7	1,000	941	997
8	1,000	938	999
9	1,000	911	997
10	1,000	906	1,000

Table 7 Comparison with Z-eigenvalue methods with data generated by uninform distribution

n	SDP	Z1	Z2
4	999	986	995
5	1,000	966	997
6	999	945	997
7	999	934	1,000
8	1,000	902	999
9	1,000	910	999
10	1,000	898	1,000

two Z-eigenvalue methods, which are referred as "Z1" and "Z2" in Tables 6 and 7. In Table 6 we generate 1,000 tensors by normal distribution, while in Table 7, the 1,000 instances are generated by uniform distribution in the interval $(-1, 1)$. We report, in Tables 6 and 7, the number of instances that are globally solved by each algorithm. According to these experiments, we can see that the performance of our approach is better than that of Z1 and is comparable to Z2.

7 What if the solution is not rank-one?

Although our numerical results strongly indicate that problems (15) and (16) are very likely to admit rank-one solutions (100 % for the randomly created problems we tested), it is in principle possible that the solution $X^* = \sum_{i=1}^{r} a^i (a^i)^\top$ is not of rank one, i.e., $r > 1$. In this situation, we can introduce a small perturbation to the original tensor $\mathcal{F}$ and apply the proposed algorithms. If the newly obtained solution is rank-one, we can say that this solution is a good approximation of the "true" solution. Another way to proceed is to apply a post-processing procedure, which will be discussed below, on X^* to obtain a rank-one solution.

We denote $\mathcal{X}^* = \boldsymbol{M}^{-1}(X^*)$, and $\mathcal{X}^* = \sum_{i=1}^{r} \mathcal{A}^i \otimes \mathcal{A}^i$, where $\boldsymbol{V}(\mathcal{A}^i) = a^i$. Basically, we want to find the projection of $\mathcal{X}^*$ onto the rank-one tensor set $\{\mathcal{X} \in \mathbf{S}^{2d} \mid \text{rank}(\mathcal{X}) = 1, \|\mathcal{X}\|_F = 1\}$:

$$\min_{\mathcal{X} \in \mathbf{S}^{2d}, \|\mathcal{X}\|_F = 1, \text{rank}(\mathcal{X}) = 1} \|\mathcal{X}^* - \mathcal{X}\|_F,$$

which is equivalent to

$$\max_{\|x^1\|=\cdots=\|x^{2d}\|=1} \mathcal{X}^* \left(x^1, x^2, \ldots, x^{2d} \right). \tag{32}$$

This is a problem in the form of (6), but the difference is that $\mathcal{X}^*$ has a further structure which plays an important role in the later discussion.

Proposition 7.1 *For a tensor* $\mathcal{F} = \sum_{i=1}^r \mathcal{A}^i \otimes \mathcal{A}^i \in \mathbf{S}^{n^{2d}}$, *it holds that*

$$\mathcal{F}\left(x^1, x^1, x^2, x^2, \ldots, x^d, x^d \right) \geq 0, \quad \forall\, x^1, x^2, \ldots, x^d. \tag{33}$$

Proof Since $\mathcal{F}$ is super-symmetric, for any $x^1, x^2, \ldots, x^d$,

$$
\begin{aligned}
\mathcal{F}\left(x^1, x^1, x^2, x^2, \ldots, x^d, x^d \right) &= \mathcal{F}\left(x^1, x^2, \ldots, x^d, x^1, x^2, \ldots, x^d \right) \\
&= \sum_{i=1}^r \mathcal{A}^i \otimes \mathcal{A}^i \left(x^1, x^2, \ldots, x^d, x^1, x^2, \ldots, x^d \right) \\
&= \sum_{i=1}^r \left(\mathcal{A}^i \left(x^1, x^2, \ldots, x^d \right) \right)^2 \geq 0.
\end{aligned}
$$

$\square$

Inequality (33) is called co-quadratic positive semidefinite. In [9], it was proved that if $\mathcal{F}$ is co-quadratic positive semidefinite then

$$\mathcal{F}\left(x^1, x^2, \ldots, x^{2d} \right) \leq \max_{1 \leq i \leq 2d} \left\{ \mathcal{F}(\underbrace{x^i, \ldots, x^i}_{2d}) \right\}.$$

As a result,

$$\max_{\|x\|=1} \mathcal{F}(\underbrace{x, \ldots, x}_{2d}) = \max_{\|x^1\|=\cdots=\|x^{2d}\|=1} \mathcal{F}\left(x^1, x^2, \ldots, x^{2d} \right). \tag{34}$$

For a solution X^*, which is either optimal to (15) or (16), by Proposition 7.1 we know that $\mathcal{X}^* = \boldsymbol{M}^{-1}(X^*)$ is co-quadratic positive semidefinite. So the problem (32) on finding the rank-one projection of $\mathcal{X}^*$ is equivalent to

$$\max_{\|x^1\|=\cdots=\|x^{2d}\|=1} \mathcal{X}^* \left(x^1, x^2, \ldots, x^{2d} \right)$$

due to equality (34). Essentially, we can resort to a multi-linear problem, which is easier than (6) and can be solved, for instance, by the MBI method proposed in [10]. Note that if we apply the MBI method directly to (6), relation (34) may not be guaranteed: the MBI may get trapped in a local minimum instead of local maximum.

8 Extensions

In this section, we show how to extend the results in the previous sections for super-symmetric tensor PCA problem to tensors that are not super-symmetric.

8.1 Bi-quadratic tensor PCA

A closely related problem to the tensor PCA problem (6) is the following bi-quadratic PCA problem:

$$
\begin{array}{ll}
\max & \mathcal{G}(x, y, x, y) \\
\text{s.t.} & x \in \mathbf{R}^n, \|x\| = 1, \\
& y \in \mathbf{R}^m, \|y\| = 1,
\end{array}
\tag{35}
$$

where $\mathcal{G}$ is a *partial-symmetric* tensor defined as follows.

Definition 8.1 A 4-th order tensor $\mathcal{G} \in \mathbf{R}^{(nm)^2}$ is called partial-symmetric if $\mathcal{G}_{ijk\ell} = \mathcal{G}_{kji\ell} = \mathcal{G}_{i\ell kj}, \forall i, j, k, \ell$. The space of all 4-th order partial-symmetric tensor is denoted by $\overrightarrow{\mathbf{S}}^{(mn)^2}$.

Various approximation algorithms for bi-quadratic PCA problem have been studied in [39]. Problem (35) arises from the strong ellipticity condition problem in solid mechanics and the entanglement problem in quantum physics; see [39] for more applications of bi-quadratic PCA problem.

We can unfold a partial-symmetric tensor $\mathcal{G}$ in a similar manner as in Definition 1.3.

Definition 8.2 For $\mathcal{G} \in \overrightarrow{\mathbf{S}}^{(nm)^2}$, we define its square matrix rearrangement, denoted by $M(\mathcal{G}) \in \mathbf{R}^{mn \times mn}$, as the following:

$$
M(\mathcal{G})_{k\ell} := \mathcal{G}_{i_1 i_2 i_3 i_4},
$$

$1 \le i_1, i_3 \le n, \ 1 \le i_2, i_4 \le m$ where $k = (i_1 - 1)m + i_2$, and $\ell = (i_3 - 1)m + i_4$.

It is easy to check that for given vectors $a \in \mathbf{R}^n$ and $b \in \mathbf{R}^m$, $a \otimes b \otimes a \otimes b \in \overrightarrow{\mathbf{S}}^{(nm)^2}$. Moreover, we say a partial-symmetric tensor $\mathcal{G}$ is of rank one if $\mathcal{G} = \lambda\, a \otimes b \otimes a \otimes b$ for some vectors a, b and scalar λ.

Since $\text{Tr}(xy^\top yx^\top) = x^\top xy^\top y = 1$, by letting $\mathcal{X} = x \otimes y \otimes x \otimes y$, problem (35) is equivalent to

$$
\begin{array}{ll}
\max & \mathcal{G} \bullet \mathcal{X} \\
\text{s.t.} & \sum_{i,j} \mathcal{X}_{ijij} = 1, \\
& \mathcal{X} \in \overrightarrow{\mathbf{S}}^{(nm)^2}, \ \text{rank}(\mathcal{X}) = 1.
\end{array}
$$

In the following, we group variables x and y together and treat $x \otimes y$ as a long vector by stacking its columns. Denote $X = M(\mathcal{X})$ and $G = M(\mathcal{G})$. Then, we end up with a matrix version of the above tensor problem:

$$\max \quad \text{Tr}(GX)$$
$$\text{s.t.} \quad \text{Tr}(X) = 1, \quad X \succeq 0,$$
$$\mathbf{M}^{-1}(X) \in \overrightarrow{\mathbf{S}}^{(nm)^2}, \quad \text{rank}(X) = 1. \tag{36}$$

As it turns out, the rank-one equivalence theorem can be extended to the partial symmetric tensors. Therefore the above two problems are actually equivalent.

Theorem 8.1 *Suppose A is an $n \times m$ dimensional matrix. Then the following two statements are equivalent:*

(i) $\text{rank}(A) = 1$;

(ii) $A \otimes A \in \overrightarrow{\mathbf{S}}^{(nm)^2}$.

In other words, suppose $\mathcal{F} \in \overrightarrow{\mathbf{S}}^{(nm)^2}$, then $\text{rank}(\mathcal{F}) = 1 \iff \text{rank}(F) = 1$, where $F = \mathbf{M}(\mathcal{F})$.

Proof (i) $\implies$ (ii) is obvious. Suppose $\text{rank}(A) = 1$, say $A = ab^\top$ for some $a \in \mathbf{R}^n$ and $b \in \mathbf{R}^m$. Then $\mathcal{G} = A \otimes A = a \otimes b \otimes a \otimes b$ is partial-symmetric.

Conversely, suppose $\mathcal{G} = A \otimes A \in \overrightarrow{\mathbf{S}}^{(nm)^2}$. Then

$$A_{i_1 j_1} A_{i_2 j_2} = \mathcal{G}_{i_1 j_1 i_2 j_2} = \mathcal{G}_{i_2 j_1 i_1 j_2} = A_{i_2 j_1} A_{i_1 j_2}, \quad \forall 1 \leq i_1, i_2 \leq n, \ 1 \leq j_1, j_2 \leq m,$$

implies $A_{i_1 j_1} A_{i_2 j_2} - A_{i_2 j_1} A_{i_1 j_2} = 0$. That is, every 2×2 minor of matrix A is zero. Thus A is of rank one. $\qquad \square$

By using the similar argument in Theorem 4.1, we can show that the following SDP relaxation of (36) has a good chance to get a low-rank solution.

$$\max \quad \text{Tr}(GX)$$
$$\text{s.t.} \quad \text{Tr}(X) = 1, \quad X \succeq 0,$$
$$\mathbf{M}^{-1}(X) \in \overrightarrow{\mathbf{S}}^{(nm)^2}. \tag{37}$$

Therefore, we used the same ADMM to solve (37). The frequency of returning rank-one solutions for randomly created examples is reported in Table 8. As in Tables 1

Table 8 Frequency of problem (37) having rank-one solution

Dim (n, m)	rank-1	CPU
(4,4)	100	0.12
(4,6)	100	0.25
(6,6)	100	0.76
(6,8)	100	1.35
(8,8)	98	2.30
(8,10)	100	3.60
(10,10)	96	5.77

and [2], we tested 100 random instances for each (n, m) and report the number of instances that produced rank-one solutions. We also report the average CPU time (in seconds) using ADMM to solve the problems. Table 8 shows that the SDP relaxation (37) can give a rank-one solution for *most* randomly created instances, and thus is likely to solve the original problem (35) to optimality.

8.2 Tri-linear tensor PCA

Now let us consider a highly non-symmetric case: tri-linear PCA.

$$
\begin{aligned}
\max \quad & \mathcal{F}(x, y, z) \\
\text{s.t.} \quad & x \in \mathbf{R}^n, \|x\| = 1, \\
& y \in \mathbf{R}^m, \|y\| = 1, \\
& z \in \mathbf{R}^\ell, \quad \|z\| = 1,
\end{aligned}
\tag{38}
$$

where $\mathcal{F} \in \mathbf{R}^{n \times m \times \ell}$ is any 3-rd order tensor and $n \le m \le \ell$.

Recently, tri-linear PCA problem was found to be very useful in many practical problems. For instance, Wang and Ahuja [56] proposed a tensor rank-one decomposition algorithm to compress image sequence, where they essentially solve a sequence of tri-linear PCA problems.

By the Cauchy–Schwarz inequality, the problem (38) is equivalent to

$$
\begin{array}{ll}
\max \quad \|\mathcal{F}(x, y, \cdot)\| & \max \quad \|\mathcal{F}(x, y, \cdot)\|^2 \\
\text{s.t.} \quad x \in \mathbf{R}^n, \|x\| = 1, \Longleftrightarrow \text{s.t.} \quad x \in \mathbf{R}^n, \|x\| = 1, \\
\quad\quad\; y \in \mathbf{R}^m, \|y\| = 1, & \quad\quad\;\; y \in \mathbf{R}^m, \|y\| = 1.
\end{array}
$$

We further notice

$$
\begin{aligned}
\|\mathcal{F}(x, y, \cdot)\|^2 = \mathcal{F}(x, y, \cdot)^\top \mathcal{F}(x, y, \cdot) &= \sum_{i,u=1}^{n} \sum_{j,v=1}^{m} \sum_{k=1}^{\ell} \mathcal{F}_{ijk} \mathcal{F}_{uvk} x_i y_j x_u y_v \\
&= \sum_{i,u=1}^{n} \sum_{j,v=1}^{m} \sum_{k=1}^{\ell} \mathcal{F}_{ivk} \mathcal{F}_{ujk} x_i y_v x_u y_j.
\end{aligned}
$$

Therefore,

$$
\|\mathcal{F}(x, y, \cdot)\|^2 = \frac{\sum_{i,u=1}^{n} \sum_{j,v=1}^{m} \left(\sum_{k=1}^{\ell} \mathcal{F}_{ijk} \mathcal{F}_{uvk} + \sum_{k=1}^{\ell} \mathcal{F}_{ivk} \mathcal{F}_{ujk} \right)}{2} x_i y_j x_u y_v,
$$

and we can find a partial symmetric tensor $\mathcal{G}$ with

$$
\mathcal{G}_{ijuv} = \sum_{k=1}^{\ell} \left(\mathcal{F}_{ijk} \mathcal{F}_{uvk} + \mathcal{F}_{ivk} \mathcal{F}_{ujk} \right) / 2, \quad \forall\, i, j, u, v,
$$

such that $\|\mathcal{F}(x, y, \cdot)\|^2 = \mathcal{G}(x, y, x, y)$. Hence, the tri-linear problem (38) can be equivalently formulated in the form of problem (35), which can be solved by the method proposed in the previous subsection.

8.3 Quadri-linear tensor PCA

In this subsection, we consider the following quadri-linear PCA problem:

$$
\begin{aligned}
\max \quad & \mathcal{F}(x^1, x^2, x^3, x^4) \\
\text{s.t.} \quad & x^i \in \mathbf{R}^{n_i}, \|x^i\| = 1, \ \forall\, i = 1, 2, 3, 4,
\end{aligned}
\tag{39}
$$

where $\mathcal{F} \in \mathbf{R}^{n_1 \times \cdots \times n_4}$ with $n_1 \leq n_3 \leq n_2 \leq n_4$. Let us first convert the quadri-linear function $\mathcal{F}(x^1, x^2, x^3, x^4)$ to a bi-quadratic function $\mathcal{T}\left(\begin{matrix} x^1 & x^2 & x^1 & x^2 \\ x^3 & x^4 & x^3 & x^4 \end{matrix}\right)$ with $\mathcal{T}$ being partial symmetric. To this end, we first construct $\mathcal{G}$ with

$$
\mathcal{G}_{i_1, i_2, n+i_3, n+i_4} = \begin{cases} \mathcal{F}_{j_1 j_2 j_3 j_4}, & \text{if } 1 \leq i_k \leq n_k \\ 0, & \text{otherwise.} \end{cases}
$$

Consequently, we have $\mathcal{F}(x^1, x^2, x^3, x^4) = \mathcal{G}\left(\begin{matrix} x^1 & x^2 & x^1 & x^2 \\ x^3 & x^4 & x^3 & x^4 \end{matrix}\right)$. Then we can further partial-symmetrize $\mathcal{G}$ and the desired tensor $\mathcal{T}$ is as follows,

$$
\mathcal{T}_{i_1 i_2 i_3 i_4} = \frac{1}{4}\left(\mathcal{G}_{i_1 i_2 i_3 i_4} + \mathcal{G}_{i_1 i_4 i_3 i_2} + \mathcal{G}_{i_3 i_2 i_1 i_4} + \mathcal{G}_{i_3 i_4 i_1 i_2}\right) \quad \forall\, i_1, i_2, i_3, i_4,
$$

satisfying $\mathcal{T}\left(\begin{matrix} x^1 & x^2 & x^1 & x^2 \\ x^3 & x^4 & x^3 & x^4 \end{matrix}\right) = \mathcal{G}\left(\begin{matrix} x^1 & x^2 & x^1 & x^2 \\ x^3 & x^4 & x^3 & x^4 \end{matrix}\right)$. Therefore, problem (39) is now reformulated as a bi-quadratic problem:

$$
\begin{aligned}
\max \quad & \mathcal{T}\left(\begin{matrix} x^1 & x^2 & x^1 & x^2 \\ x^3 & x^4 & x^3 & x^4 \end{matrix}\right) \\
\text{s.t.} \quad & x^i \in \mathbf{R}^{n_i}, \|x^i\| = 1, \ \forall\, i = 1, \ldots, 4.
\end{aligned}
\tag{40}
$$

Moreover, we can show that the above problem is actually a bi-quadratic problem in the form of (35).

Proposition 8.2 *Suppose $\mathcal{T}$ is a fourth order partial symmetric tensor. Then problem (40) is equivalent to*

$$
\begin{aligned}
\max \quad & \mathcal{T}\left(\begin{matrix} x^1 & x^2 & x^1 & x^2 \\ x^3 & x^4 & x^3 & x^4 \end{matrix}\right) \\
\text{s.t.} \quad & \sqrt{\|x^1\|^2 + \|x^3\|^2} = \sqrt{2}, \\
& \sqrt{\|x^2\|^2 + \|x^4\|^2} = \sqrt{2}.
\end{aligned}
\tag{41}
$$

Proof It is obvious that (41) is a relaxation of (40). To further prove that the relaxation (41) is tight, let $(\hat{x}^1, \hat{x}^2, \hat{x}^3, \hat{x}^4)$ be optimal to (41). Then $\mathcal{T}\begin{pmatrix} \hat{x}^1 & \hat{x}^2 & \hat{x}^1 & \hat{x}^2 \\ \hat{x}^3 & \hat{x}^4 & \hat{x}^3 & \hat{x}^4 \end{pmatrix} = \mathcal{F}(\hat{x}^1, \hat{x}^2, \hat{x}^3, \hat{x}^4) > 0$, and so $\hat{x}^i \neq 0$ for all i. Moreover, notice that

$$\sqrt{\|\hat{x}^1\|\|\hat{x}^3\|} \leq \sqrt{\frac{\|\hat{x}^1\|^2 + \|\hat{x}^3\|^2}{2}} = 1 \text{ and } \sqrt{\|\hat{x}^2\|\|\hat{x}^4\|} \leq \sqrt{\frac{\|\hat{x}^2\|^2 + \|\hat{x}^4\|^2}{2}} = 1.$$

Thus

$$\mathcal{T}\begin{pmatrix} \frac{\hat{x}^1}{\|\hat{x}^1\|} & \frac{\hat{x}^2}{\|\hat{x}^2\|} & \frac{\hat{x}^1}{\|\hat{x}^1\|} & \frac{\hat{x}^2}{\|\hat{x}^2\|} \\ \frac{\hat{x}^3}{\|\hat{x}^3\|} & \frac{\hat{x}^4}{\|\hat{x}^4\|} & \frac{\hat{x}^3}{\|\hat{x}^3\|} & \frac{\hat{x}^4}{\|\hat{x}^4\|} \end{pmatrix} = \mathcal{F}\left(\frac{\hat{x}^1}{\|\hat{x}^1\|}, \frac{\hat{x}^2}{\|\hat{x}^2\|}, \frac{\hat{x}^3}{\|\hat{x}^3\|}, \frac{\hat{x}^4}{\|\hat{x}^4\|}\right)$$

$$= \frac{\mathcal{F}(\hat{x}^1, \hat{x}^2, \hat{x}^3, \hat{x}^4)}{\|\hat{x}^1\|\|\hat{x}^2\|\|\hat{x}^3\|\|\hat{x}^4\|}$$

$$\geq \mathcal{F}(\hat{x}^1, \hat{x}^2, \hat{x}^3, \hat{x}^4)$$

$$= \mathcal{T}\begin{pmatrix} \hat{x}^1 & \hat{x}^2 & \hat{x}^1 & \hat{x}^2 \\ \hat{x}^3 & \hat{x}^4 & \hat{x}^3 & \hat{x}^4 \end{pmatrix}.$$

To summarize, we have found a feasible solution $\left(\frac{\hat{x}^1}{\|\hat{x}^1\|}, \frac{\hat{x}^2}{\|\hat{x}^2\|}, \frac{\hat{x}^3}{\|\hat{x}^3\|}, \frac{\hat{x}^4}{\|\hat{x}^4\|}\right)$ of (40), which is optimal to its relaxation (41) and thus this relaxation is tight. □

By letting $y = \begin{pmatrix} x^1 \\ x^3 \end{pmatrix}$, $z = \begin{pmatrix} x^2 \\ x^4 \end{pmatrix}$ and using some scaling technique, we can see that problem (41) share the same solution with

$$\begin{aligned} \max \quad & \mathcal{T}(y, z, y, z) \\ \text{s.t.} \quad & \|y\| = 1, \\ & \|z\| = 1, \end{aligned}$$

which was studied in Subsect. 8.1.

8.4 Even order multi-linear PCA

The above discussion can be extended to the even order multi-linear PCA problem:

$$\begin{aligned} \max \quad & \mathcal{F}(x^1, x^2, \ldots, x^{2d}) \\ \text{s.t.} \quad & x^i \in \mathbf{R}^{n_i}, \|x^i\| = 1, \ \forall \, i = 1, 2, \ldots, 2d, \end{aligned} \tag{42}$$

where $\mathcal{F} \in \mathbb{R}^{n^1 \times \cdots \times n^{2d}}$. An immediate relaxation of (42) is the following

$$\begin{aligned} \max \quad & \mathcal{F}(x^1, x^2, \ldots, x^{2d}) \\ \text{s.t.} \quad & x^i \in \mathbf{R}^{n_i}, \sqrt{\sum_i^{2d} \|x^i\|^2} = \sqrt{2d}. \end{aligned} \tag{43}$$

The following result shows that these two problems are actually equivalent.

Proposition 8.3 *It holds that problem* (42) *is equivalent to* (43).

Proof It suffices to show that relaxation (43) is tight. To this end, suppose $(\hat{x}^1, \ldots, \hat{x}^{2d})$ is an optimal solution of (43). Then $\mathcal{F}(\hat{x}^1, \hat{x}^2, \ldots, \hat{x}^{2d}) > 0$ and so $\hat{x}^i \neq 0$ for $i = 1, \ldots, 2d$. We also notice

$$\sqrt{\left(\prod_{i=1}^{2d} \|\hat{x}_i\|^2\right)^{\frac{1}{2d}}} \leq \sqrt{\sum_{i}^{2d} \|\hat{x}^i\|^2/2d} = 1.$$

Consequently, $\prod_{i=1}^{2d} \|\hat{x}_i\| \leq 1$ and

$$\mathcal{F}\left(\frac{\hat{x}^1}{\|\hat{x}^1\|}, \frac{\hat{x}^2}{\|\hat{x}^2\|}, \ldots, \frac{\hat{x}^{2d}}{\|\hat{x}^{2d}\|}\right) = \frac{\mathcal{F}(\hat{x}^1, \hat{x}^2, \ldots, \hat{x}^{2d})}{\prod_{i=1}^{2d} \|\hat{x}_i\|} \geq \mathcal{F}(\hat{x}^1, \hat{x}^2, \ldots, \hat{x}^{2d}).$$

Therefore, we have found a feasible solution $\left(\frac{\hat{x}^1}{\|\hat{x}^1\|}, \frac{\hat{x}^2}{\|\hat{x}^2\|}, \ldots, \frac{\hat{x}^{2d}}{\|\hat{x}^{2d}\|}\right)$ of (42), which is optimal to (43). This implies that the relaxation is tight. □

We now focus on (43). Based on $\mathcal{F}$, we can construct a larger tensor $\mathcal{G}$ as follows

$$\mathcal{G}_{i_1 \ldots i_{2d}} = \begin{cases} \mathcal{F}_{j_1 \ldots j_{2d}}, & \text{if } 1 + \sum_{\ell=1}^{k-1} n_\ell \leq i_k \leq \sum_{\ell=1}^{k} n_\ell \text{ and } j_k = i_k - \sum_{\ell=1}^{k-1} n_\ell \\ 0, & \text{otherwise.} \end{cases}$$

By this construction, we have

$$\mathcal{F}(x^1, x^2, \ldots, x^{2d}) = \mathcal{G}(\underbrace{y, \ldots, y}_{2d})$$

with $y = ((x^1)^\top, (x^2)^\top, \ldots, (x^{2d})^\top)^\top$. We can further symmetrize $\mathcal{G}$ and find a super-symmetric $\mathcal{T}$ such that

$$\mathcal{T}_{i_1 \ldots i_{2d}} := \frac{1}{|\pi(i_1 \ldots i_{2d})|} \sum_{j_1 \ldots j_{2d} \in \pi(i_1 \ldots i_{2d})} \mathcal{G}_{j_1 \ldots j_{2d}}, \quad \forall 1 \leq i_1, \ldots, i_{2d} \leq \sum_{\ell=1}^{2d} n_\ell,$$

and

$$\mathcal{T}(\underbrace{y, \ldots, y}_{2d}) = \mathcal{G}(\underbrace{y, \ldots, y}_{2d}) = \mathcal{F}(x^1, x^2, \ldots, x^{2d}).$$

Therefore, problem (43) is equivalent to

$$\max \quad \mathcal{T}(\underbrace{y, \ldots, y}_{2d})$$
$$\text{s.t.} \quad \|y\| = \sqrt{2d},$$

which is further equivalent to

$$\max \quad \mathcal{T}(\underbrace{y, \ldots, y}_{2d})$$
$$\text{s.t.} \quad \|y\| = 1.$$

Thus the methods we developed for solving (6) can be applied to solve (42).

8.5 Odd degree tensor PCA

The last problem studied in this section is the following odd degree tensor PCA problem:

$$\max \quad \mathcal{F}(\underbrace{x, \ldots, x}_{2d+1})$$
$$\text{s.t.} \quad \|x\| = 1, \tag{44}$$

where $\mathcal{F}$ is a $(2d + 1)$-th order super-symmetric tensor. As the degree is odd,

$$\max_{\|x\|=1} \mathcal{F}(\underbrace{x, \ldots, x}_{2d+1}) = \max_{\|x\|=1} |\mathcal{F}(\underbrace{x, \ldots, x}_{2d+1})| = \max_{\|x^i\|_2^2=1, \, i=1,\ldots,2d+1} |\mathcal{F}(x^1, \ldots, x^{2d+1})|,$$

where the last identity is due to Corollary 4.2 in [10]. The above formula combined with the fact that

$$\max_{\|x\|=1} |\mathcal{F}(\underbrace{x, \ldots, x}_{2d+1})| \leq \max_{\|x\|=1, \, \|y\|=1} |\mathcal{F}(\underbrace{x, \ldots, x}_{2d}, y)|$$
$$\leq \max_{\|x^i\|=1, \, i=1,\ldots,2d+1} |\mathcal{F}(x^1, \ldots, x^{2d+1})|$$

implies

$$\max_{\|x\|=1} \mathcal{F}(\underbrace{x, \ldots, x}_{2d+1}) = \max_{\|x\|=1, \, \|y\|=1} |\mathcal{F}(\underbrace{x, \ldots, x}_{2d}, y)| = \max_{\|x\|=1, \, \|y\|=1} \mathcal{F}(\underbrace{x, \ldots, x}_{2d}, y).$$

By using the similar technique as in Sect. 8.2, problem (44) is equivalent to an even order tensor PCA problem:

$$\max \quad \mathcal{G}(\underbrace{x, \ldots, x}_{4d})$$
$$\text{s.t.} \quad \|x\| = 1,$$

where $\mathcal{G}$ is super-symmetric with

$$\mathcal{G}_{i_1,\ldots,i_{4d}} = \frac{1}{|\pi(i_1 \ldots i_{4d})|} \sum_{k=1}^{n} \left(\sum_{j_1 \ldots j_{4d} \in \pi(i_1 \ldots i_{4d})} \mathcal{F}_{i_1 \ldots i_{2d} k} \, \mathcal{F}_{i_{2d+1} \ldots i_{4d} k} \right).$$

9 Conclusions

Tensor PCA is an emerging area of research with many important applications in image processing, data analysis, statistical learning, and bio-informatics. In this paper we introduced a new matricization scheme, which ensures that if the tensor is of rank one (in the sense of CP rank), then its matricization is a rank-one matrix, and vice versa. This enables one to apply the methodology in compressive sensing and matrix rank minimization, in particular the L_1-norm and nuclear norm optimization techniques. As it turns out, this approach is very likely to yield a rank-one solution. This effectively finds the leading PC by convex optimization, at least for randomly generated problem instances. The resulting convex optimization model is still large in general. We proposed to use the first-order method such as the ADMM method, which turns out to be very effective in this case. Multiple PCs can be computed sequentially via the so-called "deflation" technique.

References

1. Alizadeh, F.: Interior point methods in semidefinite programming with applications to combinatorial optimization. SIAM J. Optim. **5**, 13–51 (1993)
2. Bloy, L., Verma, R.: On computing the underlying fiber directions from the diffusion orientation distribution function. In: Metaxas, D., Axel, L., Fichtinger, G., Szekeley, G. (eds.) Medical Image Computing and Computer-Assisted Intervention, MICCAI (2008)
3. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**(1), 1–122 (2011)
4. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. Found. Comput. Math. **9**, 717–772 (2009)
5. Candès, E.J., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inf. Theory **52**, 489–509 (2006)
6. Candès, E.J., Tao, T.: The power of convex relaxation: near-optimal matrix completion. IEEE Trans. Inf. Theory **56**(5), 2053–2080 (2009)
7. Carroll, J.D., Chang, J.J.: Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. Psychometrika **35**(3), 283–319 (1970)
8. Chandrasekaran, V., Recht, P.A., Parrilo, B., Willsky, A.S.: The convex geometry of linear inverse problems. Found. Comput. Math. **12**(6), 805–849 (2012)
9. Chen, B.: Optimization with Block Variables: Theory and Applications. PhD thesis, The Chinese Univesrty of Hong Kong (2012)
10. Chen, B., He, S., Li, Z., Zhang, S.: Maximum block improvement and polynomial optimization. SIAM J. Optim. **22**, 87–107 (2012)
11. Comon, P., Golub, G., Lim, L.H., Mourrain, B.: Symmetric tensors and symmetric tensor rank. SIAM J. Matrix Anal. Appl. **30**(3), 1254–1279 (2008)
12. Donoho, D.: Compressed sensing. IEEE Trans. Inf. Theory **52**, 1289–1306 (2006)
13. Douglas, J., Rachford, H.H.: On the numerical solution of the heat conduction problem in 2 and 3 space variables. Trans. Am. Math. Soc. **82**, 421–439 (1956)
14. Eckstein, J.: Splitting Methods for Monotone Operators with Applications to Parallel Optimization. PhD thesis, Massachusetts Institute of Technology (1989)
15. Eckstein, J., Bertsekas, D.P.: On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math. Program. **55**, 293–318 (1992)
16. Fortin, M., Glowinski, R.: Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems. North-Holland, Amsterdam (1983)

17. Gabay, D.: Applications of the method of multipliers to variational inequalities. In: Fortin, M., Glowinski, R. (eds.) Augmented Lagrangian Methods: Applications to the Solution of Boundary Value Problems. North-Holland, Amsterdam (1983)

18. Gandy, S., Recht, B., Yamada, I.: Tensor completion and low-n-rank tensor recovery via convex optimization. Inverse Probl. **27**(2), 025010 (2011)

19. Ghosh, A., Tsigaridas, E., Descoteaux, M., Comon, P., Mourrain, B., Deriche, R.: A polynomial based approach to extract the maxima of an antipodally symmetric spherical function and its application to extract fiber directions from the orientation distribution function in diffusion MRI. In: Computational Diffusion MRI Workshop (CDMRI08), New York (2008)

20. Glowinski, R., Le Tallec, P.: Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics. SIAM, Philadelphia, PA (1989)

21. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J. ACM (JACM) **42**(6), 1115–1145 (1995)

22. Goldstein, T., Osher, S.: The split Bregman method for L1-regularized problems. SIAM J. Imaging Sci. **2**, 323–343 (2009)

23. Grant, M., Boyd, S.: CVX: Matlab Software for Disciplined Convex Programming, version 1.21. http://cvxr.com/cvx, May 2010

24. Harshman, R.A.: Foundations of the parafac procedure: models and conditions for an "explanatory" multimodal factor analysis. UCLA Working Papers in Phonetics, vol 16, pp. 1–84. http://publish.uwo.ca/~harshman/wppfac0.pdf (1970)

25. Håstad, J.: Tensor rank is NP-complete. J. Algorithms **11**, 644–654 (1990)

26. Henrion, D., Lasserre, J.B., Loefberg, J.: GloptiPoly 3: moments, optimization and semidefinite programming. Optim. Methods Softw. **24**, 761–779 (2009)

27. Hilling, J.J., Sudbery, A.: The geometric measure of multipartite entanglement and the singular values of a hypermatrix. J. Math. Phys. **51**, 072102 (2010)

28. Hitchcock, F.L.: The Expression of a Tensor or a Polyadic as a Sum of Products. J. Math. Phys. **6**, 164–189 (1927)

29. Hitchcock, F.L.: Multiple invariants and generalized rank of a p-way matrix or tensor. J. Math. Phys. **7**(1), 39–79 (1927)

30. Hu, S., Qi, L.: Algebraic connectivity of an even uniform hypergraph. J. Comb. Optim. **24**(4), 564–579 (2012)

31. Kofidis, E., Regalia, P.A.: On the best rank-1 approximation of higher-order supersymmetric tensors. SIAM J. Matrix Anal. Appl. **23**, 863–884 (2002)

32. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev. **51**, 455–500 (2009)

33. Kolda, T.G., Mayo, J.R.: Shifted power method for computing tensor eigenpairs. SIAM J. Matrix Anal. **32**, 1095–1124 (2011)

34. Kruskal, J.B.: Rank, Decomposition, and Uniqueness for 3-way and n-way arrays. In: Multiway Data Analysis, pp. 7–18 (1989)

35. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. SIAM J. Optim. **11**, 796–817 (2001)

36. Lasserre, J.B.: Polynomials nonnegative on a grid and discrete representations. Trans. Am. Math. Soc. **354**, 631–649 (2001)

37. Li, W., Ng, M.: Existence and Uniqueness of Stationary Probability Vector of a Transition Probability Tensor. Technical Report. Department of Mathematics, The Hong Kong Baptist University (2011)

38. Lim, L.H.: Singular values and eigenvalues of tensors: a variational approach. In: Computational Advances in Multi-Sensor Adaptive Processing, 2005 1st IEEE International Workshop on, pp. 129–132. IEEE (2005)

39. Ling, C., Nie, J., Qi, L., Ye, Y.: Biquadratic optimization over unit spheres and semidefinite programming relaxations. SIAM J. Optim. **20**, 1286–1310 (2009)

40. Lions, P.L., Mercier, B.: Splitting algorithms for the sum of two nonlinear operators. SIAM J. Numer. Anal. **16**, 964–979 (1979)

41. Liu, J., Musialski, P., Wonka, P., Ye, J.: Tensor completion for estimating missing values in visual data. In: The Twelfth IEEE International Conference on Computer Vision (2009)

42. Ma, S.: Alternating direction method of multipliers for sparse principal component analysis. J. Oper. Res. Soc. China **1**(2), 253–274 (2013)

43. Ma, S., Goldfarb, D., Chen, L.: Fixed point and Bregman iterative methods for matrix rank minimization. Math. Program. Ser. A **128**, 321–353 (2011)

44. Mackey, L.: Deflation methods for sparse PCA. In: Advances in Neural Information Processing Systems (NIPS) (2008)
45. Parrilo, P.A.: Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization. PhD thesis, California Institute of Technology (2000)
46. Parrilo, P.A.: Semidefinite programming relaxations for semialgebraic problems. Math. Program. Ser. B **96**, 293–320 (2003)
47. Peaceman, D.H., Rachford, H.H.: The numerical solution of parabolic elliptic differential equations. SIAM J. Appl. Math. **3**, 28–41 (1955)
48. Qi, L.: Eigenvalues of a real supersymmetric tensor. J. Symb. Comput. **40**, 1302–1324 (2005)
49. Qi, L., Wang, F., Wang, Y.: Z-eigenvalue methods for a global polynomial optimization problem. Math. Program. Ser. A **118**, 301–316 (2009)
50. Qi, L., Yu, G., Wu, E.X.: Higher order positive semi-definite diffusion tensor imaging. SIAM J. Imaging Sci. **3**(3), 416–433 (2010)
51. Recht, B., Fazel, M., Parrilo, P.: Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. SIAM Rev. **52**(3), 471–501 (2010)
52. Scheinberg, K., Ma, S., Goldfarb, D.: Sparse inverse covariance selection via alternating linearization methods. In: NIPS (2010)
53. Tao, M., Yuan, X.: Recovering low-rank and sparse components of matrices from incomplete and noisy observations. SIAM J. Optim. **21**, 57–81 (2011)
54. Tomioka, R., Suzuki, T., Hayashi, K., Kashima, H.: Statistical performance of convex tensor decomposition. In: Advances in Neural Information Processing Systems (NIPS), p. 137 (2011)
55. Vandenberghe, L., Boyd, S.: Semidefinite programming. SIAM Rev. **38**(1), 49–95 (1996)
56. Wang, H., Ahuja, N.: Compact representation of multidimensional data using tensor rank-one decomposition. In: Proceedings of the Pattern Recognition, 17th International Conference on ICPR (2004)
57. Wang, Y., Yang, J., Yin, W., Zhang, Y.: A new alternating minimization algorithm for total variation image reconstruction. SIAM J. Imaging Sci. **1**(3), 248–272 (2008)
58. Wen, Z., Goldfarb, D., Yin, W.: Alternating direction augmented Lagrangian methods for semidefinite programming. Math. Program. Comput. **2**, 203–230 (2010)
59. Yang, J., Zhang, Y.: Alternating direction algorithms for ℓ_1 problems in compressive sensing. SIAM J. Sci. Comput. **33**(1), 250–278 (2011)
60. Yuan, X.: Alternating direction methods for sparse covariance selection. J. Sci. Comput. **51**, 261–273 (2012)