Tensor Principal Component Analysis via Convex Optimization

Bo JIANG [*] Shiqian MA [†] Shuzhong ZHANG [‡]

December 10, 2012

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 yield a rank-one solution with high probability, thereby solving the original tensor PCA problem to optimality with high probability. 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.

[*]Department of Industrial and Systems Engineering, University of Minnesota, Minneapolis, MN 55455.

[†]Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, N. T., Hong Kong.

[‡]Department of Industrial and Systems Engineering, University of Minnesota, Minneapolis, MN 55455.

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:

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

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:

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

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 [46], diffusion Magnetic Resonance Imaging (MRI) [15, 2, 41], quantum entanglement problem [22], spectral hypergraph theory [23] and higher-order Markov chains [29]. This is mainly because in real life we often encounter multidimensional data, such as 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 [46] 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:

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

which is equivalent to

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

where $\otimes$ denotes the outer product between vectors, in other words,

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

The above problem is also known as the best rank-one approximation of tensor $\mathcal{A}$; cf. [25, 24]. As we shall find out later in the paper, problem (4) can be reformulated as

$$\begin{array}{ll} \max & \mathcal{F}(x, x, \cdots, x) \\ \text{s.t.} & \|x\| = 1, \end{array} \tag{5}$$

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 are introduced by Lim [30] and Qi [39] independently in 2005. Since then, various methods have been proposed to find the Z-eigenvalues [7, 40, 24, 25], which possibly correspond to some 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 $\mathcal{A} = (\mathcal{A}_{i_1 i_2 \cdots i_m})_{n_1 \times n_2 \times \cdots \times n_m}$. The space where $n_1 \times n_2 \times \cdots \times n_m$ dimensional real-valued tensor resides 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 $\mathcal{A}_{i_1 i_2 \cdots i_m}$ is invariant under any permutation of $(i_1, i_2, ..., i_m)$, i.e., $\mathcal{A}_{i_1 i_2 \cdots i_m} = \mathcal{A}_{\pi(i_1, i_2, \cdots, i_m)}$, where $\pi(i_1, i_2, \cdots, i_m)$ is any permutation of indices $(i_1, i_2, \cdots, i_m)$. 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 $\|\mathcal{A}\|_F = \sqrt{\sum_{i_1, i_2, \ldots, i_m} \mathcal{A}^2_{i_1 i_2 \cdots i_m}}$. 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

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

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

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

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

$$\mathcal{A} \bullet (x^1 \otimes \cdots \otimes x^m) := \sum_{i_1, \cdots, i_m} \mathcal{A}_{i_1, \cdots, i_m} (x^1 \otimes \cdots \otimes x^m)_{i_1, \cdots, i_m} = \sum_{i_1, \cdots, i_m} \mathcal{A}_{i_1, \cdots, i_m} \prod_{k=1}^{m} x^k_{i_k}.$$

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 not restrictive. Therefore, we will focus on the following problem of largest eigenvalue of an even order super-symmetric tensor:

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

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

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

Given any $2d$-th order super-symmetric tensor form $\mathcal{F}$, we call it *rank one* if $\mathcal{F} = \underbrace{a \otimes \cdots \otimes a}_{2d}$ for some $a \in \mathbf{R}^n$. 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* $\operatorname{rank}(\mathcal{F})$ *is the smallest integer r such that*

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

where $a_i \in \mathbf{R}^n, \lambda_i \in \mathbf{R}^1$.

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

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

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

$$\begin{array}{ll} \max & \mathcal{F} \bullet \mathcal{X} \\ \text{s.t.} & \sum\limits_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^{n} k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \cdots n^{2k_n}} = 1, \\ & \mathcal{X} \in \mathbf{S}^{n^{2d}}, \ \operatorname{rank}(\mathcal{X}) = 1, \end{array} \tag{9}$$

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

The difficulty of the above problem lies in the dealing of the rank constraint $\mathrm{rank}(\mathcal{X}) = 1$. 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 [20]. 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. [26]) 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 [25]. 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 n-th index of $\mathcal{A}$ to be the column index of resulting matrix and merge other indices of $\mathcal{A}$ into the row index of $A(n)$. The so-called n-rank of $\mathcal{A}$ is the vector with m dimension such that its n-th component corresponds to the column rank of mode-n matrix $A(n)$. The notion of n-rank has been widely used in the problems of tensor decomposition. Most recently, Liu *et al.* [33] and Gandy *et al.* [14] considered the low-n-rank tensor recovery problem, which were the first attempts to solve tensor low rank optimization problems. However, up till now, the relationship between the n-rank and CP rank is still unclear. Therefore, in the following we shall introduce a new scheme to fold 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 next section, we manage to establish some connection between the CP rank and the rank of the resulting unfolding matrix.

Definition 1.2 *For a given even-order tensor $\mathcal{F} \in \mathbf{R}^{n^{2d}}$, we define its square matrix rearrangement, denoted by $\boldsymbol{M}(\mathcal{F}) \in \mathbf{R}^{n^d \times n^d}$, as the following:*

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

where

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

Similarly we introduce below the vectorization of a tensor.

Definition 1.3 *The vectorization, $\boldsymbol{V}(\mathcal{F})$, of tensor $\mathcal{F} \in \mathbf{R}^{n^m}$ is defined as*

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

where

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

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 $\boldsymbol{M}$ and $\boldsymbol{V}$ can be defined in the same manner. In the following, we denote $X = \boldsymbol{M}(\mathcal{X})$, 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

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

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

$$\mathrm{Tr}\,(X) = \sum_{i_1,\cdots,i_d} \mathcal{X}_{i_1^2 \cdots i_d^2} = \sum_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^{n} k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \cdots n^{2k_n}}. \tag{10}$$

In light of the above discussion, if we further denote $F = \boldsymbol{M}(\mathcal{F})$, then the objective in (9) is $\mathcal{F} \bullet \mathcal{X} = \mathrm{Tr}\,(FX)$, while the first constraint $\sum\limits_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^{n} k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \cdots n^{2k_n}} = 1 \iff \mathrm{Tr}\,(X) = 1$. The hard constraint in (9) is $\mathrm{rank}(\mathcal{X}) = 1$. It is straightforward to see that if $\mathcal{X}$ is of rank one, then by letting $\mathcal{X} = \underbrace{x \otimes \cdots \otimes x}_{2d}$ and $\mathcal{Y} = \underbrace{x \otimes \cdots \otimes x}_{d}$, we have $\boldsymbol{M}(\mathcal{X}) = \boldsymbol{V}(\mathcal{Y})\boldsymbol{V}(\mathcal{Y})^\top$, which is to say that matrix $\boldsymbol{M}(\mathcal{X})$ 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 give the definition of *proper tensor* first.

Definition 2.1 *We call $\mathcal{A} \in \mathbf{R}^{n^d}$ a proper n dimensional d-th order tensor if for any index k there exits a d-tuple $\{i_1, \cdots, i_d\} \supseteq \{k\}$ such that $\mathcal{A}_{i_1,\cdots,i_d} \neq 0$.*

We have the following lemma for a proper super-symmetric tensor.

Lemma 2.1 *Suppose $\mathcal{A} \in \mathbf{R}^{n^d}$ is a proper n dimensional d-th order tensor such that $\mathcal{F} = \mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}$, i.e., $\mathcal{F}$ is super-symmetric. If $\mathcal{A}$ is also super-symmetric, then the diagonal element $\mathcal{A}_{k^d} \neq 0$ for all $1 \leq k \leq n$.*

Proof. For any given index k, suppose there is an m-tuple $(i_1 \cdots i_m)$ such that $\mathcal{A}_{i_1 \cdots i_m k^{d-m}} = 0$. For any $j_{m+1} \neq k$, we have,

$$\begin{aligned}
\mathcal{A}^2_{i_1 \cdots i_m j_{m+1} k^{d-m-1}} &= \mathcal{F}_{i_1 \cdots i_m j_{m+1} k^{d-m-1} i_1 \cdots i_m j_{m+1} k^{d-m-1}} \\
&= \mathcal{F}_{i_1 \cdots i_m k^{d-m} i_1 \cdots i_m j_{m+1} j_{m+1} k^{d-m-2}} \\
&= \mathcal{A}_{i_1 \cdots i_m k^{d-m}} \, \mathcal{A}_{i_1 \cdots i_m j_{m+1} j_{m+1} k^{d-m-2}} = 0.
\end{aligned}$$

This implies that

$$\mathcal{A}_{i_1 \cdots i_m j_{m+1} \cdots j_\ell k^{d-\ell}} = 0, \quad \forall \; 0 \leq m < \ell < d, \text{ and } j_{m+1}, \cdots, j_\ell \neq k.$$

Therefore, if there is an index k with $\mathcal{A}_{k, \cdots, k} = 0$, then $\mathcal{A}_{j_1 \cdots j_\ell k^{d-\ell}} = 0$ for all $0 < \ell < d$ and $j_1, \cdots, j_\ell \neq k$. This, combined with the assumption that $\mathcal{A}$ is a super-symmetric tensor, contradicts the fact that $\mathcal{A}$ is proper. □

We further prove the following proposition for proper tensor.

Proposition 2.2 *Suppose $\mathcal{A} \in \mathbf{R}^{n^d}$ is a proper n dimensional d-th order tensor. The following two statements are equivalent:*

$$\begin{aligned}
&(i) \quad \mathcal{A} \in \mathbf{S}^{n^d}, \;\; \textit{and} \;\; \operatorname{rank}(\mathcal{A}) = 1; \\
&(ii) \quad \mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}.
\end{aligned}$$

Proof. We shall first show (i) $\Longrightarrow$ (ii). Suppose $\mathcal{A} \in \mathbf{S}^{n^d}$ with $\operatorname{rank}(\mathcal{A}) = 1$. Then there exits a vector $a \in \mathbf{R}^n$ such that $\mathcal{A} = \underbrace{a \otimes a \otimes \cdots \otimes a}_{d}$. 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). We denote $\mathcal{F} = \mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}$. For any d-tuples $\{i_1, \cdots, i_d\}$, and one of its permutations $\{j_1, \cdots, j_d\} \in \pi(i_1, \cdots, i_d)$, it holds that

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

where the last equality is due to the fact that $\mathcal{F}$ is super-symmetric. Therefore, $\mathcal{A}$ is super-symmetric. In the following, we will prove that $\mathcal{A} \in \mathbf{R}^{n^d}$ is a rank-one tensor by induction on d. It is evident that $\mathcal{A}$ is rank-one when $d = 1$. Now we assume that $\mathcal{A}$ is rank-one when $\mathcal{A} \in \mathbf{R}^{n^{d-1}}$ and we will show that the conclusion holds when the order of $\mathcal{A}$ is d.

For $\mathcal{A} \in \mathbf{R}^{n^d}$, we already proved that $\mathcal{A}$ is super-symmetric. Since $\mathcal{A}$ is proper, by Lemma 2.1 we know that $\mathcal{A}_{k^d} \neq 0$ for all $1 \leq k \leq n$. We further observe that $\mathcal{F} \in \mathbf{S}^{n^{2d}}$ implies

$$\mathcal{A}_{i_1 \cdots i_{d-1} j} \mathcal{A}_{k^d} = \mathcal{F}_{i_1 \cdots i_{d-1} j k^d} = \mathcal{F}_{i_1 \cdots i_{d-1} k^d j} = \mathcal{A}_{i_1 \cdots i_{d-1} k} \mathcal{A}_{k^{d-1} j},$$

for any $(i_1, \cdots, i_{d-1})$. As a result,

$$\mathcal{A}_{i_1 \cdots i_{d-1} j} = \frac{\mathcal{A}_{k^{d-1} j}}{\mathcal{A}_{k^d}} \mathcal{A}_{i_1 \cdots i_{d-1} k}, \quad \forall\, j, k, (i_1, \cdots, i_{d-1}).$$

Now we can construct a vector $b \in \mathbf{R}^n$ with $b_j = \frac{\mathcal{A}_{k^{d-1} j}}{\mathcal{A}_{k^d}}$ and a tensor $\mathcal{A}(k) \in \mathbf{R}^{n^{d-1}}$ with $\mathcal{A}(k)_{i_1 \cdots i_{d-1}} = \mathcal{A}_{i_1 \cdots i_{d-1} k}$, such that

$$\mathcal{A} = b \otimes \mathcal{A}(k), \tag{11}$$

and

$$\mathcal{F} = b \otimes \mathcal{A}(k) \otimes b \otimes \mathcal{A}(k) = b \otimes b \otimes \mathcal{A}(k) \otimes \mathcal{A}(k),$$

where the last equality is due to $\mathcal{F} \in \mathbf{S}^{n^{2d}}$. On the other hand, we notice that $\mathcal{A}_{j^{d-1} k} \neq 0$ for all $1 \leq j \leq n$. This is because if this is not true then we would have

$$0 = \mathcal{A}_{j^{d-1} k} \mathcal{A}_{k^{d-1} j} = \mathcal{A}_{j^d} \mathcal{A}_{k^d},$$

which contradicts the fact that $\mathcal{A}$ is proper. This means that all the diagonal elements of $\mathcal{A}(k)$ are nonzero, implying that $\mathcal{A}(k)$ is a proper tensor. Moreover, $\mathcal{A}(k) \otimes \mathcal{A}(k) \in \mathbf{S}^{n^{2d-2}}$, because $\mathcal{F}$ is super-symmetric. Thus by induction, we can find a vector $a \in \mathbf{R}^n$ such that

$$\mathcal{A}(k) = \underbrace{a \otimes a \otimes \cdots \otimes a}_{d-1}.$$

Plugging the above into (11), we get $\mathcal{A} = b \otimes \underbrace{a \otimes a \otimes \cdots \otimes a}_{d-1}$, and thus $\mathcal{A}$ is of rank one. ☐

The following proposition shows that the result in Proposition 2.2 holds without the assumption that the given tensor is proper.

Proposition 2.3 *Suppose $\mathcal{A} \in \mathbf{R}^{n^d}$. Then the following two statements are equivalent:*

$$\begin{array}{ll} (i) & \mathcal{A} \in \mathbf{S}^{n^d}, \text{ and } \operatorname{rank}(\mathcal{A}) = 1; \\ (ii) & \mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}. \end{array}$$

Proof. (i) $\Longrightarrow$ (ii) is readily implied by the same argument in that of Proposition 2.2. To show (ii) $\Longrightarrow$ (i), it suffices to prove the result when $\mathcal{A}$ is not proper. Without loss of generality, we assume $k+1, ..., n$ are all such indices that $\mathcal{A}_{j_1 \cdots j_d} = 0$ if $\{j_1, \cdots, j_d\} \supseteq \{\ell\}$ with $k+1 \leq \ell \leq n$. Now introduce tensor $\mathcal{B} \in \mathbf{R}^{k^d}$ such that $\mathcal{B}_{i_1,\cdots,i_d} = \mathcal{A}_{i_1,\cdots,i_d}$ for any $1 \leq i_1, \cdots, i_d \leq k$. Obviously $\mathcal{B}$ is proper. Moreover, since $\mathcal{A} \otimes \mathcal{A} \in \mathbf{S}^{n^{2d}}$, it follows that $\mathcal{B} \otimes \mathcal{B} \in \mathbf{S}^{n^{2d}}$. Thanks to Proposition 2.2, there exists a vector $b \in \mathbf{R}^k$ such that $\mathcal{B} = \underbrace{b \otimes \cdots \otimes b}_{d}$. Finally, by letting $a^\top = (b^\top, \underbrace{0, \cdots, 0}_{n-k})$, we have $\mathcal{A} = \underbrace{a \otimes \cdots \otimes a}_{d}$. □

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

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

$$\operatorname{rank}(\mathcal{X}) = 1 \iff \operatorname{rank}(X) = 1.$$

Proof. As remarked earlier, that $\operatorname{rank}(\mathcal{X}) = 1 \implies \operatorname{rank}(X) = 1$ is evident. To see this, suppose $\operatorname{rank}(\mathcal{X}) = 1$ 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 $X = \boldsymbol{M}(\mathcal{X}) = \boldsymbol{V}(\mathcal{Y})\boldsymbol{V}(\mathcal{Y})^\top$, which leads to $\operatorname{rank}(X) = 1$.

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 exits $x \in \mathbf{R}^n$ such that $\boldsymbol{V}^{-1}(y) = \underbrace{x \otimes \cdots \otimes x}_{d}$ and $\mathcal{X} = \underbrace{x \otimes \cdots \otimes x}_{2d}$. □

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 1.2, is also of rank one. As a result, we can reformulate Problem (9) equivalently as the following matrix optimization problem:

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

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, 8], matrix rank minimization and low-rank matrix completion problems [42, 4, 6]. 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 works of [42, 4, 6] show that under certain randomness hypothesis of the linear operator $\mathcal{C}$, the NP-hard problem (13) is equivalent to the following nuclear norm minimization problem, which is a convex programming problem, with high probability:

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

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:

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

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 $\|\tilde{X}\|_* = \operatorname{Tr}(\tilde{X}) = 1$, 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

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

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

After solving the convex optimization problem (15) and obtaining the optimal solution $\tilde{X}$, if $\operatorname{rank}(\tilde{X}) = 1$, 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 [24].

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

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

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

Since the size of this tensor is small, we used CVX [19] 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

$$a = (0.4451, 0.1649, -0.4688, 0.1649, 0.0611, -0.1737, -0.4688, -0.1737, 0.4938)^\top.$$

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 Magnetic Resonance Imaging (MRI) application studied by Ghosh et al. in [15]. In [15], Ghosh et al. studied a fiber detection problem in diffusion Magnetic Resonance Imaging (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*

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

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

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

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).

We then conduct some numerical tests on randomly generated examples. We construct 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 is obtained by symmetrizing $\mathcal{T}$. All the numerical experiments in this paper were conducted on an Intel Core i5-2520M 2.5GHz computer with 4GB of RAM, and all the default settings of CVX were used for all the tests. We choose $d = 2$ and the dimension of $\mathcal{F}$ in the tensor PCA problem from $n = 3$ to $n = 9$. We choose $\rho = 10$. 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.

n	rank-1	CPU
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

Table 1: Frequency of nuclear norm penalty problem (15) having a rank-one solution

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:

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

Remark that replacing the rank-one constraint by SDP constraint is by now a common and standard practice; see, e.g., [1, 17, 45]. 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 nonnegative 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 $F_0 := \max\limits_{1 \leq i \leq n} \mathcal{F}_{i^{2d}}$ and v is the optimal value of the following optimization problem

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

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

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 $1 \leq i \leq n$. Moreover, by denoting $r(\rho) = |Eig^-(X^*_{PNP}(\rho))|$, 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

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

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

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

Combining (19) and (18) yields

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

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 $\mathrm{Tr}\,(F X^*_{SDR}) \geq \mathrm{Tr}\,(F X^*_{PNP})$. On the other hand, it is easy to check that for any $0 < \rho_1 < \rho_2$,

$$\mathrm{Tr}\,(F X^*_{SDR}) \leq \mathrm{Tr}\,(F X^*_{PNP}(\rho_2)) \leq \mathrm{Tr}\,(F X^*_{PNP}(\rho_1)),$$

which implies $\mathrm{Tr}\,(F X^*_{SDR}) \leq \mathrm{Tr}\,(F X^*_{PNP})$. Therefore, $\lim\limits_{\rho\to+\infty} \mathrm{Tr}\,(F X^*_{PNP}(\rho)) = \mathrm{Tr}\,(F X^*_{PNP}) = \mathrm{Tr}\,(F X^*_{SDR})$. □

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 tested, it is expected that the SDP relaxation (16) will also give rank-one solutions with high probability. 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

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

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

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 [9, 38]). 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 [32, 12, 16, 10, 11]). 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 [13]). 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 [49], compressive imaging [47, 18], robust PCA [44], sparse inverse covariance selection [50, 43], sparse PCA [34] and SDP [48] 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,

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

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:

$$\left\{\begin{array}{lll} x^{k+1} & := & \operatorname{argmin}_{x\in\mathcal{C}}\ \mathcal{L}_\mu(x,y^k;\lambda^k) \\ y^{k+1} & := & \operatorname{argmin}_{y\in\mathcal{D}}\ \mathcal{L}_\mu(x^{k+1},y;\lambda^k) \\ \lambda^{k+1} & := & \lambda^k-(Ax^{k+1}+By^{k+1}-b)/\mu, \end{array}\right. \tag{22}$$

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

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

λ is the Lagrange multiplier and $\mu>0$ is 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., [12, 10]).

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 as the form of (21), we can apply ADMM to solve them.

5.1 ADMM for Nuclear Penalty Problem (15)

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

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

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

$$\left\{ \begin{array}{lll} X^{k+1} & := & \mathrm{argmin}_{X\in\mathcal{C}} -\mathrm{Tr}\,(FY^k) + \rho\|Y^k\|_* - \langle \Lambda^k, X - Y^k\rangle + \frac{1}{2\mu}\|X - Y^k\|_F^2 \\ Y^{k+1} & := & \mathrm{argmin}\ -\mathrm{Tr}\,(FY) + \rho\|Y\|_* - \langle \Lambda^k, X^{k+1} - Y\rangle + \frac{1}{2\mu}\|X^{k+1} - Y\|_F^2 \\ \Lambda^{k+1} & := & \Lambda^k - (X^{k+1} - Y^{k+1})/\mu, \end{array} \right. \tag{24}$$

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

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

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 Section 5.2.

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

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

This problem is known to have a closed-form solution that is given by the following so-called matrix shrinkage operation (see, e.g., [35]):

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

where $U\mathrm{Diag}\,(\sigma)V^\top$ 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 $Z \in \mathbf{S}^{n^d \times n^d}$:

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

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

$$\begin{array}{ll} \min & \|\mathcal{X} - \mathcal{Z}\|_F^2 \\ \text{s.t.} & \sum\limits_{k \in \mathbb{K}(n,d)} \frac{d!}{\prod_{j=1}^n k_j!} \mathcal{X}_{1^{2k_1} 2^{2k_2} \cdots n^{2k_n}} = 1, \\ & \mathcal{X} \in \mathbf{S}^{n^{2d}}, \end{array} \tag{28}$$

where $\mathcal{X} = \boldsymbol{M}^{-1}(X)$, $\mathcal{Z} = \boldsymbol{M}^{-1}(Z)$, and the equality constraint is due to (10). Now we denote index set

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

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

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

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

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

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 of (28), the optimal solution $\mathcal{X}^*$ of Problem (28) satisfies

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

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

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

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

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

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

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

A typical iteration of ADMM for solving (30) is

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

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.,

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

where $U\mathrm{Diag}\,(\sigma)U^\top$ 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 1.2. We report in Table 3 the results on randomly created problems with $d = 2$ and $n = 7, 8, 9, 10$. 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. 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 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

Based on the results of the above tests, we may conclude that it is most efficient to solve the SDP relaxation by ADMM. In this subsection, we compare this approach with a competing method based on the Sum of Squares (SOS) approach (Lasserre [27, 28] and Parrilo [36, 37]), which can solve any general polynomial problems to any given accuracy. However, the SOS approach 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. [21] developed a specialized Matlab toolbox known as GloptiPoly 3 based on SOS approach, which will be used in our test.

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 for the same problem. 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 GloptiPoly 3 is very time consuming compared with our ADMM approach. 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 5 with $n = 16$ and instance 10 with $n = 18$).

7 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.

7.1 Bi-quadratic tensor PCA

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

$$\begin{array}{ll} \max & \mathcal{G}(x, y, x, y) \\ \text{s.t.} & x \in \mathbf{R}^n, \|x\| = 1, \\ & y \in \mathbf{R}^m, \|y\| = 1, \end{array} \tag{32}$$

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

Definition 7.1 *A 4-th order tensor $\mathcal{G} \in \mathbf{R}^{(mn)^2}$ 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 $\overrightarrow{\overrightarrow{\mathbf{S}}}^{(mn)^2}$.*

Various approximation algorithms for bi-quadratic PCA problem have been studied in [31]. Problem (32) arises from the strong ellipticity condition problem in solid mechanics and the entanglement problem in quantum physics; see [31] for more applications of bi-quadratic PCA problem.

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

Since $\mathrm{Tr}\,(xy^\top yx^\top) = x^\top xy^\top y = 1$, by letting $\mathcal{X} = x \otimes y \otimes x \otimes y$, problem (32) is equivalent to

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

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

$$\begin{array}{ll} \max & \mathrm{Tr}\,(GX) \\ \text{s.t.} & \mathrm{Tr}\,(X) = 1, \quad X \succeq 0, \\ & \boldsymbol{M}^{-1}(X) \in \overrightarrow{\overrightarrow{\mathbf{S}}}^{(mn)^2},\ \mathrm{rank}(X) = 1. \end{array} \tag{33}$$

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 7.1 *Suppose A is an $n \times m$ dimensional matrix. Then the following two statements are equivalent:*
(i) $\mathrm{rank}(A) = 1$;
(ii) $A \otimes A \ \in \overrightarrow{\overrightarrow{\mathbf{S}}}^{(mn)^2}$.
In other words, suppose $\mathcal{F} \in \overrightarrow{\overrightarrow{\mathbf{S}}}^{(mn)^2}$, then $\mathrm{rank}(\mathcal{F}) = 1 \Longleftrightarrow \mathrm{rank}(F) = 1$, *where $F = \boldsymbol{M}(\mathcal{F})$.*

Proof. (i) $\Longrightarrow$ (ii) is obvious. Suppose $\mathrm{rank}(A) = 1$, 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 $\mathcal{G} = A \otimes A \in \overrightarrow{\overrightarrow{\mathbf{S}}}^{(mn)^2}$. Without loss of generality, we can assume A to be proper (otherwise we can find a proper submatrix of A, and the whole proof goes through as well). Since A is a proper matrix, it cannot happen that $A_{jj} = 0$ for all $1 \le j \le n$. Otherwise,

$$A_{ij}A_{ji} = \mathcal{G}_{ijji} = \mathcal{G}_{jjii} = A_{jj}A_{ii} = 0,$$

where the second equality is due to $\mathcal{G}$ is partial-symmetric. For fixed j, we cannot have $A_{ji} = 0$ for all $i \neq j$ or $A_{ji} = 0$ for all $i \neq j$, which combined with $A_{jj} = 0$ contradicts the properness of A. So we must have $i, \ell \neq j$ such that $A_{\ell j}, A_{ji} \neq 0$. However, in this case,

$$0 = A_{jj}A_{\ell i} = \mathcal{G}_{jj\ell i} = \mathcal{G}_{\ell jji} = A_{\ell j}A_{ji} \neq 0$$

giving rise to a contradiction. Therefore in the following, we assume $A_{kk} \neq 0$ for some index k. Again since $\mathcal{G} \in \overrightarrow{\overrightarrow{\mathbf{S}}}^{(mn)^2}$,

$$A_{kj}A_{ik} = \mathcal{G}_{kjik} = \mathcal{G}_{ijkk} = A_{ij}A_{kk}.$$

Consequently, $\frac{A_{kj}}{A_{kk}} A_{ik} = A_{ij}$ for any $1 \le i \le n$ implies A is of rank one. □

By using the similar argument in Theorem 4.1, we can show that the following SDP relaxation of (33) has a good chance to get a low rank solution.

$$\begin{array}{ll} \max & \mathrm{Tr}\,(GX) \\ \text{s.t.} & \mathrm{Tr}\,(X) = 1, \quad X \succeq 0, \\ & \boldsymbol{M}^{-1}(X) \in \overrightarrow{\overrightarrow{\mathbf{S}}}^{(mn)^2}. \end{array} \tag{34}$$

Therefore, we used the same ADMM to solve (34). The frequency of returning rank-one solutions for randomly created examples is reported in Table 6. As in Table 1 and Table 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 6 shows that the SDP relaxation (34) can give a rank-one solution for *most* randomly created instances, and thus can solve the original problem (32) to optimality with high probability.

7.2 Tri-linear tensor PCA

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

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

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 [46] 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 (35) is equivalent to

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

We further notice

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

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

$$\mathcal{G}_{ijuv} = \sum_{k=1}^{\ell} \left(\mathcal{F}_{ijk}\, \mathcal{F}_{uvk} + \mathcal{F}_{ivk}\, \mathcal{F}_{ujk} + \mathcal{F}_{ujk}\, \mathcal{F}_{ivk} \right)/3, \quad \forall\, i, j, u, v,$$

such that $\|\mathcal{F}(x, y, \cdot)\|^2 = \mathcal{G}\,(x, y, x, y)$. Hence, the tri-linear problem (35) can be equivalently formulated in the form of problem (32), which can be solved by the method proposed in the previous subsection.

7.3 Quadri-linear tensor PCA

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

$$\begin{array}{ll} \max & \mathcal{F}(x^1, x^2, x^3, x^4) \\ \text{s.t.} & x^i \in \mathbf{R}^{n_i}, \|x^i\| = 1, \ \forall\, i = 1, 2, 3, 4, \end{array} \tag{36}$$

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

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

Consequently, we have $\mathcal{F}(x^1, x^2, x^3, x^4) = \mathcal{G}\left(\begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}, \begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}\right)$. 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 $\mathcal{T}\left(\begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}, \begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}\right) = \mathcal{G}\left(\begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}, \begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}\right)$. Therefore, problem (36) is now reformulated as a bi-quadratic problem:

$$\begin{array}{ll} \max & \mathcal{T}\left(\begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}, \begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}\right) \\ \text{s.t.} & x^i \in \mathbf{R}^{n_i}, \|x^i\| = 1, \ \forall\, i = 1, \ldots, 4. \end{array} \tag{37}$$

Moreover, we can show that the above problem is actually a bi-quadratic problem in the form of (32).

Proposition 7.2 *Suppose $\mathcal{T}$ is a fourth order partial symmetric tensor. Then problem (37) is equivalent to*

$$\begin{array}{ll} \max & \mathcal{T}\left(\begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}, \begin{smallmatrix} x^1 \\ x^3 \end{smallmatrix}, \begin{smallmatrix} x^2 \\ x^4 \end{smallmatrix}\right) \\ \text{s.t.} & \sqrt{\|x^1\|^2 + \|x^3\|^2} = \sqrt{2}, \\ & \sqrt{\|x^2\|^2 + \|x^4\|^2} = \sqrt{2}. \end{array} \tag{38}$$

Proof. It is obvious that (38) is a relaxation of (37). To further prove that the relaxation (38) is tight, we assume $(\hat{x}^1, \hat{x}^2, \hat{x}^3, \hat{x}^4)$ is optimal to (38). Then $\mathcal{T}\left(\frac{\hat{x}^1}{\hat{x}^3}, \frac{\hat{x}^2}{\hat{x}^4}, \frac{\hat{x}^1}{\hat{x}^3}, \frac{\hat{x}^2}{\hat{x}^4}\right) = \mathcal{F}(\hat{x}^1, \hat{x}^2, \hat{x}^3, \hat{x}^4) > 0$, 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

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

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 (37), which is optimal to its relaxation (38) and thus this relaxation is tight. □

By letting $y = \binom{x^1}{x^3}$, $z = \binom{x^2}{x^4}$ and using some scaling technique, we can see that problem (38) share the same solution with

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

which was studied in Subsection 7.1.

7.4 Even order multi-linear PCA

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

$$\begin{array}{ll} \max & \mathcal{F}(x^1, x^2, \cdots, x^{2d}) \\ \text{s.t.} & x^i \in \mathbf{R}^{n_i}, \|x^i\| = 1, \ \forall\, i = 1, 2, \ldots, 2d, \end{array} \tag{39}$$

where $\mathcal{F} \in \mathbb{R}^{n^1 \times \cdots \times n^{2d}}$. An immediate relaxation of (39) is the following

$$\begin{array}{ll} \max & \mathcal{F}(x^1, x^2, \cdots, x^{2d}) \\ \text{s.t.} & x^i \in \mathbf{R}^{n_i}, \sqrt{\sum\limits_i^{2d} \|x^i\|^2} = \sqrt{2d}. \end{array} \tag{40}$$

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

Proposition 7.3 *It holds that problem (39) is equivalent to (40).*

Proof. It suffices to show that relaxation (40) is tight. To this end, suppose $(\hat{x}^1, \cdots, \hat{x}^{2d})$ is an optimal solution of (40). Then $\mathcal{F}(\hat{x}^1, \hat{x}^2, \cdots, \hat{x}^{2d}) > 0$ 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

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

Therefore, we have found a feasible solution $\left(\frac{\hat{x}^1}{\|\hat{x}^1\|}, \frac{\hat{x}^2}{\|\hat{x}^2\|}, \cdots, \frac{\hat{x}^{2d}}{\|\hat{x}^{2d}\|}\right)$ of (39), which is optimal to (40) implying that the relaxation is tight. □

We now focus on (40). Based on $\mathcal{F}$, we can construct a larger tensor $\mathcal{G}$ as follows

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

By this construction, we have

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

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

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

and

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

Therefore, problem (40) is equivalent to

$$\begin{array}{ll} \max & \mathcal{T}(\underbrace{y, \cdots, y}_{2d}) \\ \text{s.t.} & \|y\| = \sqrt{2d}, \end{array}$$

which is further equivalent to

$$\begin{array}{ll} \max & \mathcal{T}(\underbrace{y,\cdots,y}_{2d}) \\ \text{s.t.} & \|y\| = 1. \end{array}$$

Thus the methods we developed for solving (6) can be applied to solve (39).

7.5 Odd degree tensor PCA

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

$$\begin{array}{ll} \max & \mathcal{F}(\underbrace{x,\cdots,x}_{2d+1}) \\ \text{s.t.} & \|x\| = 1, \end{array} \tag{41}$$

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

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

where the last identity is due to Corollary 4.2 in [7]. The above formula combined with the fact that

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

implies

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

By using the similar technique as in Subsection 7.2, problem (41) is equivalent to an even order tensor PCA problem:

$$\begin{array}{ll} \max & \mathcal{G}(\underbrace{x,\cdots,x}_{4d}) \\ \text{s.t.} & \|x\| = 1, \end{array}$$

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

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

8 Conclusions

Tensor principal component analysis 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 at the rank-one tensor (in the sense of CP rank) is equivalent to the rank-one matrix. This enables one to apply the methodology in compressive sensing, in particular the L_1-norm optimization technique. As it turns out, this approach yields a rank-one solution with a high probability. This effectively solves the tensor PCA problem 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 ADM method, which turns out to be very effective in this case.

References

[1] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5:13–51, 1993. 13

[2] L. Bloy and R. Verma. On computing the underlying fiber directions from the diffusion orientation distribution function. In *Medical Image Computing and Computer-Assisted Intervention, MICCAI 2008, D.Metaxas, L. Axel, G. Fichtinger and G. Szekeley, eds.*,, 2008. 2

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 2011. 15

[4] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9:717–772, 2009. 10

[5] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, 2006. 10

[6] E. J. Candès and T. Tao. The power of convex relaxation: near-optimal matrix completion. *IEEE Trans. Inform. Theory*, 56(5):2053–2080, 2009. 10

[7] B. Chen, S. He, Z. Li, and S. Zhang. Maximum block improvement and polynomial optimization. *SIAM Journal on Optimization*, 22:87–107, 2012. 3, 26

[8] D. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52:1289–1306, 2006. 10

[9] J. Douglas and H. H. Rachford. On the numerical solution of the heat conduction problem in 2 and 3 space variables. *Transactions of the American Mathematical Society*, 82:421–439, 1956. 15

[10] J. Eckstein. *Splitting methods for monotone operators with applications to parallel optimization.* PhD thesis, Massachusetts Institute of Technology, 1989. 15

[11] J. Eckstein and D. P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55:293–318, 1992. 15

[12] M. Fortin and R. Glowinski. *Augmented Lagrangian methods: applications to the numerical solution of boundary-value problems.* North-Holland Pub. Co., 1983. 15

[13] D. Gabay. Applications of the method of multipliers to variational inequalities. In M. Fortin and R. Glowinski, editors, *Augmented Lagrangian Methods: Applications to the Solution of Boundary Value Problems.* North-Hollan, Amsterdam, 1983. 15

[14] S. Gandy, B. Recht, and I. Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 2011. 5

[15] A. Ghosh, E. Tsigaridas, M. Descoteaux, P. Comon, B. Mourrain, and R. Deriche. 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. 2, 11

[16] R. Glowinski and P. Le Tallec. *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics.* SIAM, Philadelphia, Pennsylvania, 1989. 15

[17] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.*, 42(6):1115–1145, 1995. 13

[18] T. Goldstein and S. Osher. The split Bregman method for L1-regularized problems. *SIAM J. Imaging Sci.*, 2:323–343, 2009. 15

[19] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. *http://cvxr.com/cvx*, May 2010. 11

[20] J. Håstad. Tensor rank is NP-complete. *J. Algorithms*, 11:644–654, 1990. 5

[21] D. Henrion, J. B. Lasserre, and J. Loefberg. GloptiPoly 3: Moments, optimization and semidefinite programming. *Optimization Methods and Software*, 24:761–779, 2009. 19

[22] J. J. Hilling and A. Sudbery. The geometric measure of multipartite entanglement and the singular values of a hypermatrix. *J. Math. Phys.*, 51:072102, 2010. 2

[23] S. Hu and L. Qi. Algebraic connectivity of an even uniform hypergraph. *to appear in: Journal of Combinatorial Optimization.*, 2012. 2

[24] E. Kofidis and P. A. Regalia. On the best rank-1 approximation of higher-order supersymmetric tensors. *SIAM J. Matrix Anal. Appl.*, 23:863–884, 2002. 3, 10

[25] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51:455–500, 2009. 3, 5

[26] J. B. Kruskal. *Multiway Data Analysis*, chapter Rank, Decomposition, and Uniqueness for 3-way and $N-$way Arrays, pages 7–18. North-Holland, Amsterdam, 1989. 5

[27] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11:796–817, 2001. 19

[28] J. B. Lasserre. Polynomials nonnegative on a grid and discrete representations. *Transactions of the American Mathematical Society*, 354:631–649, 2001. 19

[29] W. Li and M. Ng. Existence and uniqueness of stationary probability vector of a transition probability tensor. Technical report, Department of Mathematics, The Hong Kong Baptist University, March 2011. 2

[30] L. H. Lim. Singular values and eigenvalues of tensors: a variational approach. In *Proceedings of the IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2005. 3

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

[32] P. L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16:964–979, 1979. 15

[33] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. In *The Twelfth IEEE International Conference on Computer Vision*, 2009. 5

[34] S. Ma. Alternating direction method of multipliers for sparse principal component analysis. *preprint*, 2011. 15

[35] S. Ma, D. Goldfarb, and L. Chen. Fixed point and Bregman iterative methods for matrix rank minimization. *Mathematical Programming Series A*, 128:321–353, 2011. 16

[36] P. A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, 2000. 19

[37] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming, Series B*, 96:293–320, 2003. 19

[38] D. H. Peaceman and H. H. Rachford. The numerical solution of parabolic elliptic differential equations. *SIAM Journal on Applied Mathematics*, 3:28–41, 1955. 15

[39] L. Qi. Eigenvalues of a real supersymmetric tensor. *Journal of Symbolic Computation*, 40:1302–1324, 2005. 3

[40] L. Qi, F. Wang, and Y. Wang. Z-eigenvalue methods for a global polynomial optimization problem. *Mathematical Programming, Series A*, 118:301–316, 2009. 3

[41] L. Qi, G. Yu, and E. X. Wu. Higher order positive semi-definite diffusion tensor imaging. *SIAM Journal on Imaging Sciences*, pages 416–433, 2010. 2

[42] B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010. 10

[43] K. Scheinberg, S. Ma, and D. Goldfarb. Sparse inverse covariance selection via alternating linearization methods. In *NIPS*, 2010. 15

[44] M. Tao and X. Yuan. Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM J. Optim.*, 21:57–81, 2011. 15

[45] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Rev.*, 38(1):49–95, 1996. 13

[46] H. Wang and N. Ahuja. Compact representation of multidimensional data using tensor rank-one decomposition. In *Proceedings of the Pattern Recognition, 17th International Conference on ICPR*, 2004. 2, 22

[47] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008. 15

[48] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented Lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, 2:203–230, 2010. 15

[49] J. Yang and Y. Zhang. Alternating direction algorithms for ℓ_1 problems in compressive sensing. *SIAM Journal on Scientific Computing*, 33(1):250–278, 2011. 15

[50] X. Yuan. Alternating direction methods for sparse covariance selection. *Journal of Scientific Computing*, 51:261–273, 2012. 15

Inst. #	Nuclear Norm Penalty (15)				SDP (16)			
	Sol.Dif.	Val.Dif.	T_{ADMM}	T_{CVX}	Sol.Dif.	Val.Dif.	T_{ADMM}	T_{CVX}
Dimension $n = 7$								
1	4.10e-005	1.89e-007	1.00	32.66	7.05e-005	4.20e-006	0.79	2.44
2	1.62e-004	4.19e-006	0.91	35.32	1.49e-004	2.31e-008	0.64	2.63
3	1.50e-004	5.46e-006	1.00	36.78	9.33e-005	4.46e-006	0.61	2.49
4	1.65e-004	4.05e-006	0.98	33.57	7.15e-005	4.17e-006	0.63	2.48
5	4.77e-005	4.36e-006	0.95	32.05	1.36e-005	4.27e-006	0.65	2.40
6	7.07e-005	4.64e-006	0.95	31.17	7.83e-005	4.32e-006	0.62	2.40
7	5.31e-005	4.55e-006	0.87	28.72	2.42e-005	3.89e-006	0.64	2.42
8	5.77e-005	3.26e-007	0.88	38.33	1.40e-004	8.58e-007	0.67	2.38
9	1.53e-004	4.29e-006	0.95	32.39	6.07e-005	3.85e-007	0.68	2.43
10	6.15e-005	4.13e-006	0.98	33.29	2.58e-005	1.97e-007	0.69	2.43
Dimension $n = 8$								
1	3.90e-005	4.23e-006	1.87	180.08	1.72e-004	3.39e-008	1.31	4.86
2	8.31e-005	5.06e-006	1.85	183.90	1.09e-004	5.44e-006	1.15	4.77
3	5.58e-005	4.97e-006	1.79	176.69	3.47e-005	7.18e-008	1.27	5.93
4	4.23e-005	4.13e-006	1.82	170.36	9.11e-005	9.36e-008	1.21	4.49
5	9.12e-005	2.02e-007	1.69	151.61	9.27e-005	9.51e-007	1.12	4.51
6	3.94e-005	4.14e-006	1.67	185.01	1.59e-004	4.93e-006	1.20	4.66
7	2.88e-005	5.32e-006	1.74	151.82	2.46e-005	4.65e-006	1.17	4.50
8	7.70e-005	4.36e-006	1.77	176.79	5.30e-005	1.31e-007	1.24	4.70
9	7.62e-005	2.11e-006	1.46	199.26	3.47e-004	8.41e-007	0.47	4.62
10	1.38e-005	4.88e-006	1.70	167.18	7.62e-005	4.60e-006	1.18	4.78
Dimension $n = 9$								
1	6.80e-005	4.74e-006	3.42	692.29	3.71e-004	1.51e-007	2.47	8.77
2	1.61e-004	4.65e-006	3.40	736.90	1.57e-004	4.77e-006	2.19	8.70
3	1.47e-004	4.71e-006	3.33	672.06	1.52e-004	1.25e-007	2.30	8.83
4	4.92e-005	1.33e-007	3.02	668.35	1.19e-004	3.42e-007	2.31	8.89
5	5.94e-005	4.86e-006	3.26	703.02	5.78e-005	1.57e-007	2.26	8.85
6	1.11e-004	5.29e-006	3.40	789.86	5.54e-005	5.95e-006	2.29	8.70
7	6.59e-005	1.01e-007	3.29	683.18	1.08e-004	6.15e-007	2.28	8.98
8	9.82e-005	6.07e-007	2.23	728.11	1.67e-004	7.70e-007	0.94	9.19
9	1.28e-004	5.32e-006	3.27	750.91	9.81e-005	4.60e-007	1.96	8.97
10	5.00e-005	5.09e-006	3.33	611.53	8.89e-005	7.15e-007	2.17	8.81
Dimension $n = 10$								
1	3.14e-005	5.66e-006	6.15	3092.51	2.55e-005	5.88e-006	4.17	22.18
2	7.14e-005	5.78e-006	6.02	2906.35	9.97e-005	5.78e-007	3.81	23.58
3	3.94e-005	5.17e-006	6.11	3065.38	4.05e-005	5.44e-006	4.21	24.01
4	7.33e-005	6.15e-006	6.04	3042.79	9.31e-005	1.79e-007	4.33	23.96
5	8.91e-005	5.34e-006	6.30	3158.84	5.80e-005	5.81e-006	4.36	26.68
6	3.77e-005	6.01e-006	6.10	2933.19	8.58e-005	5.69e-006	4.36	22.62
7	2.05e-004	2.90e-007	6.40	2908.58	8.87e-005	1.38e-008	4.02	30.01
8	8.48e-005	6.63e-006	6.14	3927.01	1.02e-004	3.68e-008	4.07	27.78
9	3.73e-005	6.42e-006	6.05	3086.98	5.88e-005	1.47e-007	4.06	22.86
10	6.81e-005	6.11e-006	6.08	3380.49	1.91e-004	6.45e-006	3.91	24.82

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

Inst. #	SDP				NNP		
	Sol.Dif.	Val.Dif.	T_{ADMM}	T_{CVX}	Sol.Dif.$_{DS}$	Val.Dif.$_{DS}$	T_{ADMM}
Dimension $n = 14$							
1	1.76e-004	8.26e-006	31.95	188.98	1.76e-004	8.77e-006	53.06
2	4.37e-004	2.35e-007	30.86	187.38	4.37e-004	8.17e-006	44.70
3	2.40e-004	8.19e-006	28.42	192.68	2.40e-004	8.12e-006	48.06
4	1.24e-004	8.17e-006	32.94	198.02	1.24e-004	8.28e-006	51.91
5	3.87e-004	8.16e-006	34.90	189.34	3.87e-004	8.04e-006	48.87
6	4.53e-005	8.63e-006	24.86	201.61	4.49e-005	8.39e-006	50.92
7	6.71e-004	8.38e-006	30.25	193.43	6.71e-004	8.65e-006	51.50
8	1.61e-004	7.34e-006	32.05	198.15	1.61e-004	4.95e-007	47.14
9	9.40e-005	8.43e-006	37.79	199.00	9.41e-005	8.18e-006	54.02
10	1.67e-004	7.44e-006	31.60	191.67	1.67e-004	5.40e-007	47.96
Dimension $n = 16$							
1	4.03e-005	8.74e-006	68.11	732.87	4.05e-005	9.29e-006	147.47
2	1.72e-004	9.31e-006	71.91	684.05	1.71e-004	9.76e-006	180.08
3	5.38e-005	9.07e-006	91.38	687.36	5.39e-005	9.79e-006	166.88
4	1.46e-004	9.26e-006	79.85	689.29	1.46e-004	2.87e-008	166.12
5	8.27e-005	8.85e-006	75.19	682.30	8.22e-005	3.59e-007	181.18
6	3.17e-005	9.35e-006	80.15	703.35	3.17e-005	9.34e-006	179.90
7	5.66e-005	1.88e-007	82.60	696.51	5.73e-005	9.17e-006	171.28
8	2.30e-004	2.56e-007	78.83	702.43	2.30e-004	2.89e-007	167.25
9	9.24e-005	9.48e-006	77.83	735.89	9.26e-005	9.79e-006	152.90
10	3.37e-004	9.79e-006	79.17	706.87	3.37e-004	2.67e-007	155.83
Dimension $n = 18$							
1	1.09e-004	3.34e-007	104.48	2254.16	1.09e-004	2.61e-007	220.93
2	4.41e-004	1.13e-005	158.64	2016.83	4.41e-004	8.69e-008	281.94
3	1.55e-004	2.68e-007	191.92	2249.16	1.55e-004	1.97e-007	338.29
4	1.85e-004	1.09e-005	177.42	2004.44	1.85e-004	4.31e-007	295.31
5	5.03e-005	1.07e-005	169.82	2079.49	5.03e-005	1.07e-005	266.01
6	1.37e-004	1.29e-007	166.35	2039.80	1.37e-004	4.09e-007	260.92
7	4.13e-004	1.09e-005	167.65	2054.42	4.13e-004	1.09e-005	280.23
8	2.85e-004	1.39e-006	48.50	2383.20	2.85e-004	1.08e-005	159.33
9	5.89e-005	1.00e-005	163.10	2259.23	5.95e-005	1.07e-005	270.30
10	1.49e-004	1.07e-005	173.84	2187.49	1.49e-004	1.02e-005	290.24
Dimension $n = 20$							
1	5.58e-004	6.70e-007	389.27	5978.72	5.58e-004	1.21e-005	644.93
2	2.30e-004	1.33e-007	350.43	6040.70	2.30e-004	1.54e-007	636.87
3	3.66e-004	1.23e-005	372.13	5245.99	3.66e-004	1.19e-005	649.96
4	1.17e-004	2.89e-007	357.42	5529.12	1.18e-004	1.19e-005	689.26
5	5.58e-004	6.70e-007	387.78	5575.93	5.58e-004	1.21e-005	641.90
6	8.80e-005	1.52e-007	385.76	6284.73	8.88e-005	1.17e-005	641.87
7	1.31e-004	1.22e-005	388.05	5812.31	1.30e-004	1.22e-007	726.95
8	4.55e-004	1.18e-005	367.04	5305.20	4.55e-004	2.17e-008	629.76
9	5.88e-004	1.40e-007	389.02	5186.03	5.88e-004	1.96e-007	670.42
10	4.47e-004	2.38e-007	359.13	5559.39	4.47e-004	1.14e-005	695.63

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

Inst. #	GLP			SDP by ADMM		
	Val.	Time	Status	Val.	Time	Sol.R.
Dimension $n = 14$						
1	5.59	108.65	1	5.59	29.18	1
2	5.83	90.36	1	5.83	32.53	1
3	5.18	116.07	1	5.18	9.88	1
4	5.32	101.22	1	5.32	30.82	1
5	5.82	99.11	1	5.82	30.86	1
6	5.71	108.86	1	5.71	30.57	1
7	5.91	108.60	1	5.91	28.94	1
8	5.69	110.22	1	5.69	31.03	1
9	6.89	83.20	1	6.89	31.49	1
10	5.96	103.31	1	5.96	29.58	1
Dimension $n = 16$						
1	6.53	434.68	1	6.53	93.02	1
2	6.65	358.54	1	6.65	102.13	1
3	6.28	401.87	1	6.28	92.15	1
4	6.58	421.67	1	6.58	92.17	1
5	6.26	383.33	0	6.26	92.12	1
6	6.80	429.70	1	6.80	86.28	1
7	6.00	453.46	1	6.00	79.87	1
8	6.73	314.64	1	6.72	93.27	1
9	6.34	383.75	1	6.34	79.20	1
10	6.66	381.94	1	6.66	92.71	1
Dimension $n = 18$						
1	6.65	1279.10	1	6.65	152.64	1
2	6.37	1619.29	1	6.37	84.49	1
3	6.97	1393.22	1	6.97	155.34	1
4	7.00	1418.60	1	7.00	197.82	1
5	7.49	1289.53	1	7.49	173.35	1
6	6.99	1463.83	1	6.99	178.05	1
7	6.77	1476.79	1	6.77	180.71	1
8	6.87	1426.56	1	6.87	165.64	1
9	6.97	1363.82	1	6.97	176.70	1
10	6.73	1317.64	0	6.73	206.81	1
Dimension $n = 20$						
1	7.40	8614.12	1	7.40	317.03	1
2	7.33	9203.73	1	7.33	357.46	1
3	7.21	8548.82	1	7.21	374.20	1
4	7.32	7866.87	1	7.32	364.95	1
5	7.28	8641.94	1	7.28	347.89	1
6	7.02	6725.36	0	7.02	382.33	1
7	7.40	12273.17	1	7.40	389.29	1
8	6.93	13298.42	1	6.93	328.74	1
9	7.33	12274.62	1	7.33	363.64	1
10	6.68	8411.78	1	6.68	386.33	1

Table 5: Comparison between GloptiPoly 3 and the SDP Relaxation by ADMM with $\mu = 0.5$.

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

Table 6: Frequency of problem (34) having rank-one solution