# Efficient Algorithm for Sparse Tensor-variate Gaussian Graphical Models via Gradient Descent

**Pan Xu**
University of Virginia

**Tingting Zhang**
University of Virginia

**Quanquan Gu**
University of Virginia

## Abstract

We study the sparse tensor-variate Gaussian graphical model (STGGM), where each way of the tensor follows a multivariate normal distribution whose precision matrix has sparse structures. In order to estimate the precision matrices, we propose a sparsity constrained maximum likelihood estimator. However, due to the complex structure of the tensor-variate GGMs, the likelihood based estimator is non-convex, which poses great challenges for both computation and theoretical analysis. In order to address these challenges, we propose an efficient alternating gradient descent algorithm to solve this estimator, and prove that, under certain conditions on the initial estimator, our algorithm is guaranteed to linearly converge to the unknown precision matrices up to the optimal statistical error. Experiments on both synthetic data and real world brain imaging data corroborate our theory.

## 1 INTRODUCTION

High-dimensional tensor data are ubiquitous in many research fields such as computer vision (Vasilescu and Terzopoulos, 2002), recommendation systems (Xiong et al., 2010) and neuroscience (Rendle and Schmidt-Thieme, 2010; Allen, 2012; Zhou et al., 2013), to name a few. For example, functional magnetic resonance imaging (fMRI) data are naturally represented by three-way tensors. Traditional statistical and computational methods are insufficient to analyze these tensor-valued data due to their ultrahigh dimensionality as well as complex structures. This motivates tensor-based statistical and machine learning methods (Vasilescu and Terzopoulos, 2002; Kolda and Bader, 2009; Xiong et al., 2010; Rendle and Schmidt-Thieme, 2010; Allen, 2012; Zhou et al., 2013; He et al., 2014), which are able to harness the power of tensor representation. In our study, we consider the estimation of conditional independence structure within tensor data. For example, in fMRI data analysis, one aims to estimate the functional connectivity in terms of dependency structure across different regions or even voxels of the brain. One straightforward way is to vectorize the tensor data and estimate a single precision matrix (i.e., inverse covariance matrix) of the Gaussian graphical model (Friedman et al., 2008; Ravikumar et al., 2011; Rothman et al., 2008; Wang et al., 2016a) using the vectorized data. However, such an approach ignores the tensor structure and requires estimating a huge precision matrix, which is computationally expensive or even intractable.

To address the above problem, the tensor-variate Gaussian graphical model (TGGM) has been proposed by He et al. (2014) to encode the structure of tensor data. In particular, a $K$-th order (a.k.a., $K$-way) tensor $\mathcal{T} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_K}$ follows the tensor normal distribution with zero mean and covariance matrices $\mathbf{\Sigma}_1^*, \ldots, \mathbf{\Sigma}_K^*$, denoted by $\mathcal{T} \sim TN(\mathbf{0}; \mathbf{\Sigma}_1^*, \ldots, \mathbf{\Sigma}_K^*)$, if its probability density function is

$$p(\mathcal{T}|\mathbf{\Sigma}_1^*, \ldots, \mathbf{\Sigma}_K^*) = \frac{1}{\sqrt{(2\pi)^m}} \prod_{k=1}^{K} |\mathbf{\Sigma}_k^*|^{-\frac{m}{2m_k}} \exp\Big( - \frac{\|\mathcal{T} \times (\mathbf{\Sigma}^*)^{-\frac{1}{2}}\|_F^2}{2} \Big), \tag{1.1}$$

where $|\mathbf{\Sigma}_k^*|$ is the determinant of $\mathbf{\Sigma}_k^*$, $m = \prod_{k=1}^K m_k$ and $(\mathbf{\Sigma}^*)^{-\frac{1}{2}} = \{(\mathbf{\Sigma}_1^*)^{-\frac{1}{2}}, \ldots, (\mathbf{\Sigma}_K^*)^{-\frac{1}{2}}\}$. An important property of the tensor-variate Gaussian graphical model is that the covariance matrix of the tensor normal distribution is separable in the sense that it is the Kronecker product of small covariance matrices, each of which corresponds to one way of the tensor. This substantially reduces the degree of freedom of the model and makes the model estimation computationally more tractable. More importantly, it enables

Proceedings of the $20^{th}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54. 

model estimation with even one tensor sample (Sun et al., 2015).

In this paper, we aim to estimate the unknown precision matrices $\mathbf{\Omega}_k^* = (\mathbf{\Sigma}_k^*)^{-1}$ for $k = 1, \ldots, K$ given $n$ observations $\mathcal{T}_1, \ldots, \mathcal{T}_n$, which are sampled identically and independently from the tensor normal distribution in (1.1). Following He et al. (2014); Sun et al. (2015), we assume the precision matrices are sparse, i.e., the number of nonzero entries in $\mathbf{\Omega}_k^*$ satisfies $\|\mathbf{\Omega}_k^*\|_{0,0} = s_k^*$, for $k = 1, \ldots, K$. The sparse precision matrix of each way measures the conditional independence among the unfolded tensor data for that way. The resulting model is referred to as a *Sparse Tensor-variate Gaussian Graphical Model* (STGGM) (He et al., 2014; Sun et al., 2015). We propose a sparsity constrained maximum log-likelihood based estimator for the sparse precision matrices. Since the corresponding negative log-likelihood function is not jointly convex with respect to the precision matrices, and the sparsity constraints are nonconvex, it is both computationally and theoretically challenging to solve the above estimation problem. To address these challenges, we propose an efficient alternating gradient descent algorithm to solve it. In particular, our algorithm alternatively minimizes the non-convex objective function with respect to each individual precision matrix while fixing the others, under a sparsity constraint on that precision matrix.

We prove that, when the initial solutions are sufficiently close to the unknown precision matrices, our algorithm is guaranteed to linearly converge to the unknown precision matrices up to optimal statistical error. In particular, the estimator from our algorithm for the sparse precision matrix of $k$-th way attains $O_P(\sqrt{m_k s_k^* \log m_k/(nm)})$ statistical convergence rate in terms of Frobenius norm, where $s_k^*$ is the sparsity of $\mathbf{\Omega}_k^*$. It is minimax optimal since this is the best rate one can obtain even when the rest $K-1$ precision matrices are known (Cai et al., 2016; Sun et al., 2015). Thorough experiments on both synthetic and real-world neuroimaging datasets validate the superiority of our algorithm over the state-the-art algorithms.

The remainder of this paper is organized as follows: In Section 2, we briefly review existing works that are relevant to our study. We present the algorithm in Section 3, and the main theory in Section 4. In Section 5, we compare the proposed algorithm with existing algorithms on both synthetic data and real-world brain imaging data. Finally, we conclude this paper in Section 6.

**Notation** We denote the index set $\{1, \ldots, K\}$ by $[K]$. For a pair of matrices $\mathbf{A}, \mathbf{B}$ with commensurate dimensions, we let $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{tr}(\mathbf{A}^\top \mathbf{B})$ denote the inner product between $\mathbf{A}$ and $\mathbf{B}$, and let $\mathbf{A} \otimes \mathbf{B}$ denote the Kronecker product between them. For a matrix $\mathbf{A} \in \mathbb{R}^{m \times m}$, we denote by $\mathrm{vec}(\mathbf{A})$ the vectorization of $\mathbf{A}$, which converts $\mathbf{A}$ into a column vector. For a square matrix $\mathbf{A}$, we denote by $\mathbf{A}^{-1}$ the inverse of $\mathbf{A}$, and denote by $|\mathbf{A}|$ its determinant. We use the notation $\|\cdot\|$ for various types of matrix norms, including the induced norm $\|\mathbf{A}\|_p = \sup_{\|\mathbf{x}\|_p = 1} \|\mathbf{A}\mathbf{x}\|_p$ for $0 < p < \infty$, and the Frobenius norm $\|\mathbf{A}\|_F = \sqrt{\sum_{i,j=1}^m A_{ij}^2}$. Also we have $\|\mathbf{A}\|_{0,0} = \sum_{i,j} \mathbb{1}(A_{ij} \neq 0)$, $\|\mathbf{A}\|_{\infty,\infty} = \max_{1 \leq i,j \leq m} |A_{ij}|$, $\|\mathbf{A}\|_{1,1} = \sum_{i,j=1}^m |A_{ij}|$, where $\mathbb{1}(\cdot)$ is the indicator function. For a symmetric matrix $\mathbf{A}$, we use $\mathbf{A} \succ \mathbf{0}$ to denote that $\mathbf{A}$ is positive definite.

## 2 RELATED WORK

In this section, we briefly review the existing work that are relevant to ours.

When $K = 1$, the sparse tensor-variate Gaussian graphical model reduces to the sparse GGM (Friedman et al., 2008; Ravikumar et al., 2011; Rothman et al., 2008), which has been widely studied in the literature. When $K = 2$, the model STGGM reduces to the sparse matrix-variate Gaussian graphical model (MGGM), which has been studied by Leng and Tang (2012); Yin and Li (2012); Tsiligkaridis et al. (2013); Ning and Liu (2013); Zhou et al. (2014); Chen and Liu (2015). Existing studies (Leng and Tang, 2012; Yin and Li, 2012; Kalaitzis et al., 2013; Tsiligkaridis et al., 2013; Ning and Liu, 2013) have been devoted to developing various penalized maximum likelihood approaches for estimating the sparse precision matrices in MGGM. Nevertheless, all these results are stated in the sense that there exists a local minimizer that enjoys certain good statistical properties. Moreover, most statistical results require the sample size $n$ goes to infinity at a certain rate, which is in fact not necessary. The only exceptions are Zhou et al. (2014); Chen and Liu (2015). However, neither of them have any theoretical guarantee for their optimization algorithms.

For the sparse tensor-variate graphical model, He et al. (2014) showed the existence of a local optimum with desired statistical convergence rates, but did not provide a practical algorithm that is able to achieve such a good local optimum. The most related work to ours is Sun et al. (2015), which proposed an alternating minimization algorithm, and proved the optimal statistical rate of convergence for the estimators returned by the algorithm. However, their algorithm is based on exact minimization in each iteration, which is not efficient enough. Furthermore, they assume that the true pre-

cision matrices satisfy $\|\mathbf{\Omega}_k^*\|_F = 1$ for all $k = 1, \ldots, K$, which can not be achieved at the same time by normalizing the data for $K \geq 2$ in practice. In sharp contrast, our proposed algorithm achieves the same statistical rate under much milder conditions, as will be discussed later in Section 4.

Another line of related work to ours is nonconvex optimization, which has been widely used in practice due to its superior empirical performance. Very recently, alternating optimization has been analyzed for low-rank matrix estimation (Jain et al., 2013; Hardt, 2014; Zhao et al., 2015; Zheng and Lafferty, 2015; Chen and Wainwright, 2015; Bhojanapalli et al., 2015; Gu et al., 2016; Park et al., 2016; Wang et al., 2016b, 2017), sparse coding (Arora et al., 2015), phase retrieval (Netrapalli et al., 2013; Candes et al., 2015), expectation maximization (EM) algorithm (Balakrishnan et al., 2014; Zhao et al., 2015), to mention a few. However, none of these algorithms and theories can be directly extended to precision matrix estimation in sparse tensor Gaussian graphical models when $K \geq 2$, due to the complex structure of STGGM.

# 3 THE PROPOSED ESTIMATOR AND ALGORITHM

## 3.1 Tensor Algebra

Throughout our analysis, we follow the tensor notations in Kolda and Bader (2009). We denote higher order tensors by $\mathcal{T}$. Note that a $K$-th order tensor $\mathcal{T} \in \mathbb{R}^{m_1 \times \ldots \times m_K}$ reduces to a vector when $K = 1$, and reduces to a matrix when $K = 2$. The $(i_1, \ldots, i_K)$-th element of $\mathcal{T}$ is denoted as $\mathcal{T}_{i_1,\ldots,i_K}$. The vectorization of $\mathcal{T}$ is denoted by $\mathrm{vec}(\mathcal{T}) := (\mathcal{T}_{1,1,\ldots,1}, \ldots, \mathcal{T}_{m_1,1,\ldots,1}, \ldots, \mathcal{T}_{m_1,m_2,\ldots,m_K})^\top \in \mathbb{R}^m$ with $m = \prod_{k=1}^K m_k$. Fibers are the higher-order analogue of matrix rows and columns. A fiber of the tensor data is obtained by fixing every index but one, and thus the mode-$k$ fiber of $\mathcal{T}$ is denoted as $\mathcal{T}_{i_1,\ldots,i_{k-1},:,i_{k+1},\ldots,i_K}$. Matricization refers to the operation that reorders the elements of a $K$-way array into a matrix, which is sometimes called as unfolding or flattening. The mode-$k$ matricization of $\mathcal{T}$ is denoted by $\mathcal{T}_{(k)}$, resulting a matrix whose columns are the mode-$k$ fibers. We also introduce the tensor multiplication here, the $k$-mode (matrix) product of a tensor $\mathcal{T} \in \mathbb{R}^{m_1 \times \ldots \times m_K}$ with a matrix $\mathbf{A} \in \mathbb{R}^{J \times m_k}$ is denoted by $\mathcal{T} \times_k \mathbf{A}$, which is of size $m_1 \times \ldots m_{k-1} \times J \times m_{k+1} \times \ldots \times m_K$. We have $\big[\mathcal{T} \times_k \mathbf{A}\big]_{(k)} = \mathbf{A}\mathcal{T}_{(k)}$. In addition, for a list of matrices $\{\mathbf{A}_1, \ldots, \mathbf{A}_K\}$ with $\mathbf{A}_k \in \mathbb{R}^{m_k \times m_k}, k = 1, \ldots, K$, we define $\mathcal{T} \times \{\mathbf{A}_1, \ldots, \mathbf{A}_K\} := \mathcal{T} \times_1 \mathbf{A}_1 \times_2 \ldots \times_K \mathbf{A}_K$.

## 3.2 Estimator

Given i.i.d. tensor samples $\mathcal{T}_1, \ldots, \mathcal{T}_n$ from the tensor normal distribution $TN(\mathbf{0}; \mathbf{\Sigma}_1^*, \ldots, \mathbf{\Sigma}_K^*)$, our goal is to estimate the unknown sparse precision matrices $(\mathbf{\Omega}_1^*, \ldots, \mathbf{\Omega}_K^*)$ which satisfy $\|\mathbf{\Omega}_k^*\|_{0,0} = s_k^*$ for $k \in [K]$. We employ the maximum likelihood principle to estimate $\mathbf{\Omega}_k^*$ for $k \in [K]$, which minimizes the negative log-likelihood of (1.1) over $n$ samples. According to He et al. (2014); Kolda and Bader (2009), it can be shown that $\mathcal{T} \sim TN(\mathbf{0}; \mathbf{\Sigma}_1^*, \ldots, \mathbf{\Sigma}_K^*)$ if and only if $\mathrm{vec}(\mathcal{T}) \sim \mathcal{N}(\mathrm{vec}(\mathbf{0}), \mathbf{\Sigma}_K^* \otimes \cdots \otimes \mathbf{\Sigma}_1^*)$, where $\mathrm{vec}(\mathbf{0}) \in \mathbb{R}^m$ and $\otimes$ is the matrix Kronecker product. Therefore, the negative log-likelihood is equivalent to the following sample loss function up to a constant

$$q_n(\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_K) = \frac{1}{m} \mathrm{tr}\big[\widehat{\mathbf{\Sigma}}\big(\mathbf{\Omega}_K \otimes \cdots \otimes \mathbf{\Omega}_1\big)\big] - \sum_{k=1}^K \frac{1}{m_k} \log|\mathbf{\Omega}_k|, \tag{3.1}$$

where $\widehat{\mathbf{\Sigma}} = 1/n \sum_{i=1}^n \mathrm{vec}(\mathcal{T}_i)\mathrm{vec}(\mathcal{T}_i)^\top$, and $|\mathbf{\Omega}_k|$ is the determinant of $\mathbf{\Omega}_k$. We propose an estimator based on sparsity constrained maximum likelihood estimation as follows

$$\begin{aligned} &\min_{\mathbf{\Omega}_1,\ldots,\mathbf{\Omega}_K \succ \mathbf{0}} \quad q_n(\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_K) \\ &\text{subject to } \|\mathbf{\Omega}_k\|_{0,0} \leq s_k, k = 1, \ldots, K, \end{aligned} \tag{3.2}$$

where $\mathbf{\Omega}_k \succ \mathbf{0}$ means that $\mathbf{\Omega}_k$ is positive definite and $s_k$ is a tuning parameter that controls the sparsity of $\mathbf{\Omega}_k$. As will be seen in our theory, $s_k$ needs to be larger than the unknown sparsity $s_k^*$ in order to achieve a statistically good estimator. In practice, $s_k$ can be chosen by cross-validation, or a held-out set.

## 3.3 Algorithm

The loss function in (3.2) is not jointly convex with respect to $(\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_K)$. However, it is convex with respect to $\mathbf{\Omega}_k$ while fixing the rest $K-1$ matrices. This is referred to as the biconvex property in optimization. According to this property, we propose to solve the non-convex problem by alternatively updating one precision matrix with other matrices fixed. The detailed algorithm is displayed in Algorithm 1.

We require the initial points $\widehat{\mathbf{\Omega}}_k^{(0)}$ in Algorithm 1 lie in the small neighborhood of $\mathbf{\Omega}_k^*$, which is defined as the Frobenius norm ball for each $\mathbf{\Omega}_k^*$ as follows: $\mathbb{B}_F(\mathbf{\Omega}_k^*, r) = \{\mathbf{\Omega} \in \mathbb{R}^{m_k \times m_k} : \|\mathbf{\Omega} - \mathbf{\Omega}_k^*\|_F \leq r\}$, $k = 1, \ldots, K$. In practice, these initial points can be obtained by heuristic random initialization, or by the precision matrix estimator of each mode from existing graphical Lasso method (Friedman et al., 2008; Rothman et al., 2008; Ravikumar et al., 2011) on the unfolded data.

**Algorithm 1** Alternating Gradient Descent (AltGD) for STGGM

1: **Input:** Function $q_n(\mathbf{\Omega}_{[K]})$, max number of iterations $T$, initial points $\widehat{\mathbf{\Omega}}_k^{(0)} \in \mathbb{B}_F(\mathbf{\Omega}_k^*, r)$, $t = 0$, sparsity $s_k > 0$, tensor samples $\mathcal{T}_1, \ldots, \mathcal{T}_n$.
2: **for** $t = 1$ to $T$ **do**
3: **for** $k = 1$ to $K$ **do**
4: $\widehat{\mathbf{\Omega}}_k^{(t+0.5)} = \widehat{\mathbf{\Omega}}_k^{(t)} - \eta_k \nabla_k q_n\big(\widehat{\mathbf{\Omega}}_k^{(t)}, \widehat{\mathbf{\Omega}}_{[K]-k}^{(t)}\big)$;
5: $\widehat{\mathbf{\Omega}}_k^{(t+1)} = \mathrm{trunc}\Big(\widehat{\mathbf{\Omega}}_k^{(t+0.5)}, \mathcal{S}_k^{(t+0.5)}\Big)$, where $\mathcal{S}_k^{(t+0.5)}$ is the support set of the largest $s_k$ magnitudes of $\widehat{\mathbf{\Omega}}_k^{(t+0.5)}$;
6: **end for**
7: **end for**
8: **output:** $\widehat{\mathbf{\Omega}}_1^{(T)}, \ldots, \widehat{\mathbf{\Omega}}_K^{(T)}$.

There are two layers of loops in Algorithm 1. The outer loop is the iteration of gradient descent, while the inner loop is the operation over each mode of tensor data. In the inner loop, we estimate $\mathbf{\Omega}_k$ sequentially, while fixing all the other precision matrices, by solving the following sparsity constrained optimization

$$\min_{\mathbf{\Omega}_k \succ \mathbf{0}} \quad q_n(\mathbf{\Omega}_k, \mathbf{\Omega}_{[K]-k}) \quad \text{subject to } \|\mathbf{\Omega}_k\|_{0,0} \leq s_k, \tag{3.3}$$

where $[K]-k$ denotes the set of index $\{1, \ldots, k-1, k+1, \ldots, K\}$. Instead of solving (3.3) using exact optimization, we propose to perform one-step gradient descent for $\mathbf{\Omega}_k$, which corresponds to Line 4 of Algorithm 1, where $\nabla_k q_n(\mathbf{\Omega}_k, \mathbf{\Omega}_{[K]-k})$ denotes the gradient of $q_n(\mathbf{\Omega}_{[K]})$ with respect to $\mathbf{\Omega}_k$, while fixing the other $K-1$ precision matrices.

Since we require $\mathbf{\Omega}_k$ to be sparse, i.e., $\|\mathbf{\Omega}_k\|_{0,0} \leq s_k$, we apply a truncation step right after the gradient descent step for $\mathbf{\Omega}_k$, in Line 5 of Algorithm 1. The truncation step is defined as follows: for a matrix $\mathbf{\Omega} \in \mathbb{R}^{m \times m}$ and a tuple set $\mathcal{S} \subseteq \{(i,j) : i,j = 1, \ldots, m\}$, $\mathrm{trunc}(\mathbf{\Omega}, \mathcal{S})$ gives a $m \times m$ matrix, whose entries are calculated as follows

$$[\mathrm{trunc}(\mathbf{\Omega}, \mathcal{S})]_{ij} = \begin{cases} \Omega_{ij} & \text{if } (i,j) \in \mathcal{S} \\ 0 & \text{if } (i,j) \notin \mathcal{S}. \end{cases} \tag{3.4}$$

## 4 MAIN THEORY

In this section, we present our main theory which characterizes the convergence rate of our proposed algorithm and the statistical rate of the estimator.

In order to simplify our analysis, we revise Algorithm 1 into the re-sampling version by employing sample splitting technique (Hansen, 2000; Balakrishnan et al., 2014; Wang et al., 2014), which is stated in Algorithm 2. Given $n$ samples and maximum number of iterations $T$, the key idea is to split the whole dataset into $T$ pieces and use a fresh piece of data of size $\lfloor n/T \rfloor$ in each iteration. In the rest of this section, we are going to analyze Algorithm 2.

**Algorithm 2** AltGD with Sample Splitting

1: **Input:** Function $q_n(\mathbf{\Omega}_{[K]})$, max number of iterations $T$, initial points $\widehat{\mathbf{\Omega}}_k^{(0)} \in \mathbb{B}_F(\mathbf{\Omega}_k^*, r)$, $t = 0$, sparsity $s_k > 0$, tensor samples $\mathcal{T}_1, \ldots, \mathcal{T}_n$ which are split into $T$ subsets of size $\lfloor n/T \rfloor$.
2: **for** $t = 1$ to $T$ **do**
3: **for** $k = 1$ to $K$ **do**
4: $\widehat{\mathbf{\Omega}}_k^{(t+0.5)} = \widehat{\mathbf{\Omega}}_k^{(t)} - \eta_k \nabla_k q_{n/T}\big(\widehat{\mathbf{\Omega}}_k^{(t)}, \widehat{\mathbf{\Omega}}_{[K]-k}^{(t)}\big)$, which is calculated on the $t$-th data subset;
5: $\widehat{\mathbf{\Omega}}_k^{(t+1)} = \mathrm{trunc}\Big(\widehat{\mathbf{\Omega}}_k^{(t+0.5)}, \mathcal{S}_k^{(t+0.5)}\Big)$, where $\mathcal{S}_k^{(t+0.5)}$ is the support set of the largest $s_k$ magnitudes of $\widehat{\mathbf{\Omega}}_k^{(t+0.5)}$;
6: **end for**
7: **end for**
8: **output:** $\widehat{\mathbf{\Omega}}_1^{(T)}, \ldots, \widehat{\mathbf{\Omega}}_K^{(T)}$.

We first lay out an assumption that is required throughout our analysis.

**Assumption 4.1.** For any $k = 1, \ldots, K$, there is a constant $\nu > 0$ such that

$$0 < \frac{1}{\nu} \leq \lambda_{\min}(\mathbf{\Sigma}_k^*) \leq \lambda_{\max}(\mathbf{\Sigma}_k^*) \leq \nu < \infty,$$

where $\lambda_{\min}(\mathbf{\Sigma}_k^*)$ and $\lambda_{\max}(\mathbf{\Sigma}_k^*)$ are the minimal and maximal eigenvalues of $\mathbf{\Sigma}_k^*$ respectively.

Assumption 4.1 requires the uniform bound on the eigenvalues of true covariance matrices $\mathbf{\Sigma}_k^*$. This assumption is commonly imposed in the literature for the analysis of graphical models (Ravikumar et al., 2011; He et al., 2014; Sun et al., 2015). Note that since $\mathbf{\Omega}_k^* = (\mathbf{\Sigma}_k^*)^{-1}$, by the property of eigenvalues for inverse matrix, we immediately obtain $1/\nu \leq \lambda_{\min}(\mathbf{\Omega}_k^*) \leq \lambda_{\max}(\mathbf{\Omega}_k^*) \leq \nu$.

Now we are ready to present our main theory.

**Theorem 4.2.** Suppose Assumption 4.1 holds. Define $R_{\min} = \min_k \|\mathbf{\Omega}_k^*\|_F$ and $R_{\max} = \max_k \|\mathbf{\Omega}_k^*\|_F$. Let $r = \min\{1/(2\nu), R_{\min}/2, \sqrt{m}/(3Km_k\nu^{K+2}) - 2R_{\max}\}$ and suppose that the initial points satisfy $\widehat{\mathbf{\Omega}}_k^{(0)} \in \mathbb{B}_F(\mathbf{\Omega}_k^*, r)$ for all $k = 1, \ldots, K$. In Algorithm 2, let $\eta_k = 8m_k\nu^2/(16\nu^4 + 1)$ be the step size and $T > 0$ be the maximum number of iterations. Suppose the truncation parameter $s_k$ satisfies

$$\max\left\{36, \frac{16}{(1/\rho - 1)^2}\right\} \cdot s_k^* \leq s_k \leq C_1 R_{\min}^2 \cdot \frac{nmm_k}{T\nu^4 \log m_k},$$

where $C_1 > 0$ is an absolute constant. We define $\rho$ and $\tau$ as

$$\rho = 1 - \frac{2[\sqrt{m} - 6Km_k(r/2 + R_{\max})\nu^{K+2}]}{\sqrt{m}(16\nu^4 + 1)},$$
$$\tau = \frac{C_2\nu^2}{16\nu^4 + 1}\sqrt{\frac{Tm_k s_k^* \log m_k}{nm}},$$

where $C_2 > 0$ is an absolute constant. Then for any $k = 1, \ldots, K$, we have with probability at least $1 - 4m_k^2 \exp\{-nm/(2Tm_k)\}$ that

$$\big\|\widehat{\boldsymbol{\Omega}}_k^{(t)} - \boldsymbol{\Omega}_k^*\big\|_F \leq \frac{\tau}{1 - \sqrt{\rho}} + \sqrt{\rho^t} \cdot r. \tag{4.1}$$

In Theorem 4.2, we require the initial points to be close to $\boldsymbol{\Omega}_k^*$, that is, $\widehat{\boldsymbol{\Omega}}_k^{(0)} \in \mathbb{B}_F(\boldsymbol{\Omega}_k^*, r)$, where $r$ is the radius of the balls. This can be achieved by various strategies, as discussed in Section 3.3.

**Remark 4.3.** In (4.1) of Theorem 4.2, the first term is the statistical error, and the second term is the optimization error. The statistical error of the estimator returned by our algorithm is $O_P(\sqrt{m_k s_k^* \log m_k/(nm)})$. This matches the minimax lower bound when the rest $K - 1$ precision matrices are known (Cai et al., 2016; Sun et al., 2015). Therefore, our statistical rate is optimal. In order to ensure $\rho < 1$, we assume that $r \leq \sqrt{m}/(3Km_k\nu^{K+2}) - 2R_{\max}$. Recall that $R_{\max} = \max_k \|\boldsymbol{\Omega}_k^*\|_F$ and $\nu$ is the upper bound of the largest eigenvalue of $\boldsymbol{\Sigma}_k^*$, are constants, we can always find a small enough $r$ via an appropriate initialization algorithm. This enables linear convergence rate for the optimization error. Specifically, when $T$ is chosen to be no less than $C\log(nm/(m_k s_k^*))$, where $C > 0$ is a constant, the optimization error becomes smaller than the statistical error, which makes the total estimation error optimal.

An direct implication of Theorem 4.2 is that, when $K \geq 2$ and the dimensions $m_k$'s are of the same order of magnitude, the estimator by our algorithm is consistent even there is only one observation, i.e., $n = 1$. This is consistent with the best known algorithms for sparse MGGM (Zhou et al., 2014; Chen and Liu, 2015) and STGGM (Sun et al., 2015).

**Remark 4.4.** The statistical rate of our algorithm is the same as Sun et al. (2015). However, as will be shown in the experiments, our algorithm is more efficient and achieves higher accuracy than Sun et al. (2015). The reason is that our algorithm performs gradient descent rather than exact minimization in each iteration. This saves the computational cost. Moreover, their analysis relies on the assumption that $\|\boldsymbol{\Omega}_k^*\|_F = 1$ for $k = 1, \ldots, K$, which is often not true in practice and can not be achieved for all $k$ by normalization when $K \geq 2$.

# 5 EXPERIMENTS

In this section, we present numerical results on both synthetic and real world datasets to verify the performance of our algorithm, and compare it with the state-of-the-art methods.

## 5.1 Baseline Algorithms

We compare our approach (AltGD) in Algorithm 1 with the following three baseline methods: (1) the $k$-nearest neighbors $k$-NN classifier with $k = 1$ using Euclidean distance; (2) sparse Gaussian graphical model (SGGM) for vectorized tensor data, where we choose QUIC[1] algorithm to solve the graphical Lasso estimator; (3) sparse tensor-variate Gaussian graphical model (STGGM) (Sun et al., 2015) with an algorithm, denoted by Tlasso, which uses alternating minimization to estimate the precision matrices.

Specifically, in the simulation experiment in Section 5.2, we compare the performance on precision matrices estimation of our algorithm AltGD with that of Tlasso. And in the real world data experiment, we compare the performances of all methods on classification since the true precision matrices are unknown, we will discuss the details in Section 5.3 and 5.4. Since both Tlasso and AltGD require good initialization points, we adopt random definite positive matrices $\widehat{\boldsymbol{\Omega}}_1^{(0)}, \ldots, \widehat{\boldsymbol{\Omega}}_K^{(0)}$ as the initial points. We found that random initialization works very well in our experiments. Note that the $k$-NN method does not provide estimation for parameters, and the SGGM method does not provide the estimation for every individual precision matrix on each mode. Therefore they are not compared in the synthetic data experiment.

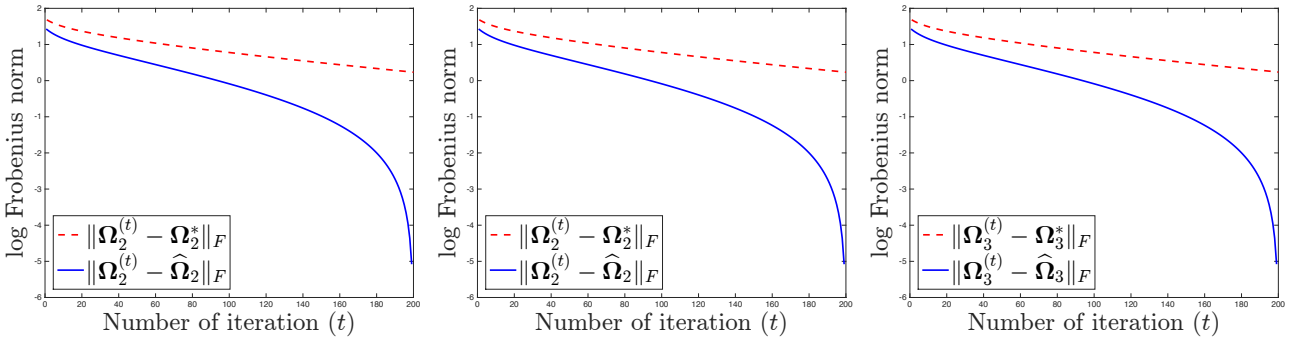
## 5.2 Synthetic Data

In the synthetic experiment, we considered three 3rd order tensor data of different dimensions: (1) $m_1 = 10, m_2 = 15, m_3 = 20$, (2) $m_1 = 25, m_2 = 25, m_3 = 25$, and (3) $m_1 = 10, m_2 = 10, m_3 = 100$. We set $s_k^* = 3m_k$, for $k = 1, 2, 3$. For example, in Setting (1), the true sparsity levels of $\boldsymbol{\Omega}_1, \boldsymbol{\Omega}_2, \boldsymbol{\Omega}_3$ are $30\%, 20\%, 15\%$ respectively. We first generated the precision matrix $\boldsymbol{\Omega}_k$ by **huge** package (Zhao et al., 2012), and we chose 'hub' pattern as the graph structure. Then the vectorized tensor samples $\mathrm{vec}(\mathcal{T}_1), \ldots, \mathrm{vec}(\mathcal{T}_n)$ were generated following multivariate normal distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_3^* \otimes \boldsymbol{\Sigma}_2^* \otimes \boldsymbol{\Sigma}_1^*)$, where $\boldsymbol{\Sigma}_k^* = (\boldsymbol{\Omega}_k^*)^{-1}$ and sample size $n = 100$. Finally, recall the tensor algebra in Section 3.1, we used the inverse transformation of vectorization to obtain the tensor samples $\mathcal{T}_1, \ldots, \mathcal{T}_n \in \mathbb{R}^{m_1 \times m_2 \times m_3}$.

[1] `http://www.cs.utexas.edu/~sustik/QUIC`

Table 1: Estimation errors of precision matrices in terms of Frobenius norm on the synthetic datasets.

| | | $\|\mathbf{\Omega}_1^{(T)} - \mathbf{\Omega}_1^*\|_F$ | $\|\mathbf{\Omega}_2^{(T)} - \mathbf{\Omega}_2^*\|_F$ | $\|\mathbf{\Omega}_3^{(T)} - \mathbf{\Omega}_3^*\|_F$ | Time $(s)$ |
|---|---|---|---|---|---|
| Setting 1 | Tlasso | 3.5915±0.0026 | 3.8643±0.0040 | 4.7183±0.0049 | 22.63 |
| | AltGD | 2.8669±0.0043 | 3.3456±0.0512 | 3.6398±0.0225 | 7.84 |
| Setting 2 | Tlasso | 2.1823±0.0215 | 2.1665±0.0361 | 2.1635±0.0248 | 79.16 |
| | AltGD | 1.7004±0.0041 | 1.7044±0.0055 | 1.7009±0.0059 | 40.74 |
| Setting 3 | Tlasso | 4.4356±0.0011 | 4.4355±0.0007 | 12.1631±0.0028 | 68.70 |
| | AltGD | 3.2393±0.0180 | 3.2549±0.0044 | 9.9892±0.0241 | 47.05 |




(a) (b) (c)

Figure 1: The logarithm of the estimation and optimization errors of $\mathbf{\Omega}_k^*$, $k = 1, 2, 3$, in terms of Frobenius norm under Setting (2): $m_1 = 25, m_2 = 25, m_3 = 25$.

The regularization parameter $\lambda$ in Tlasso, and the truncation parameters $s_k$'s in our AltGD were tuned by grid search. The step sizes $\eta_k$'s of our algorithm are chosen by line search. The best results were reported for each method. Both methods were run with maximum iteration $T = 200$. Figure 1 shows the logarithm of the estimation errors $\|\mathbf{\Omega}_k^{(t)} - \mathbf{\Omega}_k^*\|_F$ and optimization errors $\|\mathbf{\Omega}_k^{(t)} - \widehat{\mathbf{\Omega}}_k\|_F$ versus number of iteration $t$ under Setting (2). The optimization error of our algorithm decays linearly (after taken logarithm) to zero. This confirms the linear convergence rate of our algorithm. In addition, the statistical error of our estimator converges to a value larger than zero, which dominates the total estimation error. We did not plot the estimation errors versus number of iterations for Tlasso since it employs an exact minimization. In Figure 2 we also plotted $\|\widehat{\mathbf{\Omega}}_1^{(T)} - \mathbf{\Omega}_1^*\|_F$ against the scaled statistical error $\sqrt{m_1 s_1^* \log m_1/(nm)}$, which exhibits a linear dependency and validates our theory. In Table 1 we summarize the performances on precision matrix estimation between our algorithm AltGD and the Tlasso. We report the mean and standard error of estimation errors over 10 replications in Frobenius norm, where we used different samples in each replication. From Table 1, it can be seen that, in all settings, our method AltGD gives smaller errors than Tlasso in the estimation of precision matrix on every mode. Furthermore, our AltGD is also more efficient in terms of running time.

We included more experiment results for other settings in the longer version of this paper.

### 5.3 Stimulus Classification Based on fMRI Data

In this experiment, we test our algorithm on fMRI data. Since fMRI data are high-dimensional tensor data, we apply our method to investigate patterns of subject '04820' in the StarPlus fMRI experiment[2] (Mitchell et al., 2004).

During the fMRI data recording, the subject was asked

[2] `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-81/www/`

Table 2: Comparison of methods in terms of classification accuracy (%) and training time (in second) on the fMRI datasets. Note that $k$-NN method does not have a training time.

| Dataset | Data 1 | | Data 2 | | Data 3 | |
|---|---|---|---|---|---|---|
| | Accuracy | Time ($s$) | Accuracy | Time ($s$) | Accuracy | Time ($s$) |
| $k$-NN | 58.40±7.04 | - | 48.80±9.10 | - | 46.00±5.42 | - |
| SGGM (QUIC) | 77.64±10.15 | 0.02 | 65.40±6.13 | 0.01 | 66.60±9.09 | 0.02 |
| STGGM (Tlasso) | 78.42±8.37 | 3.07 | 65.00±8.07 | 3.45 | 68.80±10.01 | 4.80 |
| STGGM (AltGD) | 88.40±3.98 | 0.15 | 70.40±6.72 | 0.14 | 73.20±5.83 | 0.15 |

Table 3: Comparison of methods in terms of classification accuracy (%) and training time (in second) on the EEG datasets. Note that $k$-NN method does not have a training time.

| Dataset | Data 1 | | Data 2 | | Data 3 | |
|---|---|---|---|---|---|---|
| | Accuracy | Time ($s$) | Accuracy | Time ($s$) | Accuracy | Time ($s$) |
| $k$-NN | 41.00±5.89 | - | 65.00±6.72 | - | 52.88±5.64 | - |
| SGGM (QUIC) | 73.33±6.67 | 16.74 | 80.63±9.19 | 15.45 | 69.09±11.37 | 18.64 |
| STGGM (Tlasso) | 80.67±7.17 | 41.90 | 92.81±3.62 | 47.05 | 74.24±4.23 | 33.36 |
| STGGM (AltGD) | 82.33±4.73 | 18.86 | 93.50±3.29 | 16.59 | 76.21±4.63 | 28.99 |

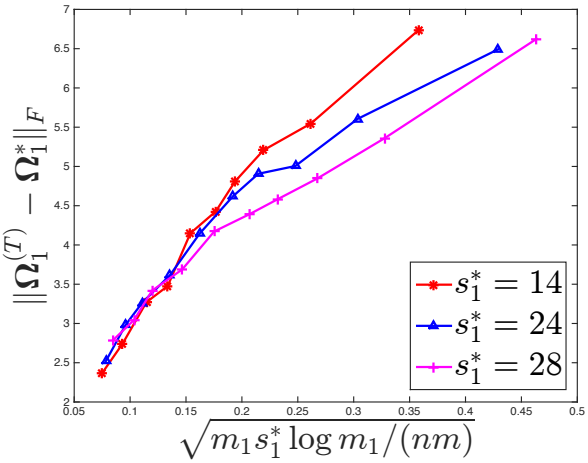



Figure 2: Scaled estimation errors for $\boldsymbol{\Omega}_1^*$ in Frobenius norm. We fixed the dimension as Setting (1): $m_1 = 10, m_2 = 15, m_3 = 20$ and varied the sparsity $s_1^*$.

to perform 40 tasks. In each task, the subject was shown an image and a sentence. The sentence either explained the image—for example, an image of a star with a plus above is paired with the sentence "The plus is above the star"—or negated the image. We used the subject's fMRI measurements of $5,015$ voxels sampled on a $64 \times 64 \times 8$ grid in the brain across $54 - 55$ time points. We generated dataset 1 by choosing two different stimuli from the same subject in this data, each consisting of 55 (samples size) tensors with dimension $6 \times 8 \times 3$ extracted from the $64 \times 64 \times 8$ grid. Moreover, we classified the tensors into two classes according to the stimulus, so each class contains 55 samples which are assumed to follow tensor-valued normal distribution. Dataset 2 and 3 were generated in the same way but from different subjects.

Since the true precision matrices of the real tensor data are unknown, we use classification as a surrogate to measure the estimation performance. For each of the two stimuli ($i = 1$ or 2) we define the following discriminant function, which is proportional to the negative log likelihood function of TGGM:

$$\delta_i(\mathcal{T}) = \mathrm{vec}(\mathcal{T})^\top (\boldsymbol{\Omega}_1^i \otimes \ldots \otimes \boldsymbol{\Omega}_K^i)\mathrm{vec}(\mathcal{T}) - \sum_{k=1}^{K} \log |\boldsymbol{\Omega}_k^i|, \tag{5.1}$$

where $\mathcal{T}$ is a $K$-order tensor sample and $\boldsymbol{\Omega}_1^i, \ldots, \boldsymbol{\Omega}_K^i$ are the estimated precision matrices based on training data from class $i$. We classify each tensor sample $\mathcal{T}$ to the class $i$ associated with the smaller $\delta_i(\mathcal{T})$, where the two classes correspond to different stimuli from the same subject. Note that for classification analysis of the SGGM model, we plug the estimated precision matrix for vectorized tensor data into the discriminant function to replace the Kronecker product.

For each dataset, we randomly partitioned the samples into the training (80%) and testing (20%) sets. We repeated the partition process 10 times and calculated the mean and standard deviation of the classification accuracy over the 10 repetitions. In each repetition,

the regularization parameters in SGGM method and Tlasso method were tuned by 3-fold cross validation on the training set. The truncation parameters $s_k$'s in AltGD were also tuned by 3-fold cross validation on the training set. The step size of AltGD is chosen by line search. Tlasso and AltGD were run with $T = 40$ iterations. Table 2 summarizes the mean and standard error of classification accuracies of different methods over 10 repetitions. We also report the average training time over 10 repetitions for different methods except $k$-NN, which does not have training time.

Table 2 shows that our proposed algorithm outperforms the other methods in terms of classification accuracy and training time. Note that QUIC algorithm for SGGM is also very fast but achieves much worse accuracy. The reason is that our method takes into account the precision matrix structure on each mode, and this information is totally lost in $k$-NN and sparse SGGM method. In addition, our algorithm is able to achieve a linear rate of convergence and attain the minimax optimal statistical rate according to our main theory. Our algorithm AltGD also outperforms Tlasso even though they achieve the same statistical rate (while under different conditions). This is probably due to the very strong assumptions in their analysis, whereas our approach requires a much milder condition.

### 5.4 EEG Data of Motor Imagery

In this experiment, we applied the proposed algorithm to the EEG datasets from the database[3]. These datasets were recorded from several healthy subjects. The cue-based BCI paradigm consisted of two motor imagery tasks: the imagination of movement of the left hand (LH) and right hand (RH). Within each trial, subjects were presented with visual stimuli while their brain activities (in $\mu V$ ) were measured at 256 Hz for 1 second. We used three subsets of the EEG data, with two classes (i.e., tasks) in each datasets: one class for LH observations and the other for RH. Each class in three datasets has 75, 81, and 165 samples respectively. For dataset 1, the time series data were converted into $64 \times 24$ spectrograms using short-time Fourier transform with Hamming window of length 64, 34 overlapping samples, leading to 3-mode tensors of size $6 \times 64 \times 24$. The spectrograms were normalized across samples to have zero means and unit variances. Similar operations were taken on datasets 2 and 3, leading to 3-mode tensors of size $6 \times 64 \times 32$ with 81 samples for each class, and $6 \times 64 \times 24$ tensors with 165 samples each class, respectively.

We used the same discriminant function (5.1) as in the fMRI experiment to evaluate different methods' estimation performance. The same as in the previous experiment, we randomly partitioned the samples in each dataset into the training (80%) and the testing data (20%). We repeated the partition 10 times and calculated the mean and standard deviation of classification accuracy over the 10 repetitions. In each repetition, the regularization parameters in SGGM method as well as Tlasso method were tuned by 3-fold cross validation. The truncation parameters $s_k$'s in AltGD were also tuned by 3-fold cross validation on the training sets. The step size of AltGD is chosen by line search. Tlasso and AltGD were run with $T = 40$ iterations.

Table 3 summarizes the experimental results for EEG data. Again, the proposed algorithm outperforms the other methods in terms of classification accuracy and training time, which is consistent with previous experimental results on fMRI data. This again validates the superior performance of our algorithm.

[3] `http://www.bsp.brain.riken.jp/~qibin/homepage/Datasets.html`

## 6 CONCLUSIONS

We proposed a sparsity constrained maximum likelihood estimator for STGGM, and develop an efficient alternating gradient descent algorithm for solving the nonconvex optimization problem corresponding to the estimator. Despite the nonconvexity of the optimization problem under study, we prove that the proposed algorithm achieves a linear convergence rate to the unknown precision matrices up to the optimal statical error. Experiments on both synthetic and real brain imaging data further support our theory.

#### Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This research was sponsored in part by the National Science Foundation IIS-1618948 and IIS-1652539. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

## References

Allen, G. (2012). Sparse higher-order principal components analysis. In *International Conference on Artificial Intelligence and Statistics*.

Arora, S., Ge, R., Ma, T. and Moitra, A. (2015). Simple, efficient, and neural algorithms for sparse coding. *arXiv preprint arXiv:1503.00778* .

Balakrishnan, S., Wainwright, M. J. and Yu, B. (2014). Statistical guarantees for the em algorithm:

From population to sample-based analysis. *arXiv preprint arXiv:1408.2156* .

Bhojanapalli, S., Kyrillidis, A. and Sanghavi, S. (2015). Dropping convexity for faster semi-definite optimization. *arXiv preprint* .

Cai, T. T., Liu, W., Zhou, H. H. et al. (2016). Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation. *The Annals of Statistics* **44** 455–488.

Candes, E. J., Li, X. and Soltanolkotabi, M. (2015). Phase retrieval via wirtinger flow: Theory and algorithms. *Information Theory, IEEE Transactions on* **61** 1985–2007.

Chen, X. and Liu, W. (2015). Statistical inference for matrix-variate gaussian graphical models and false discovery rate control. *arXiv preprint arXiv:1509.05453* .

Chen, Y. and Wainwright, M. J. (2015). Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025* .

Friedman, J., Hastie, T. and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9** 432–441.

Gu, Q., Wang, Z. and Liu, H. (2016). Low-rank and sparse structure pursuit via alternating minimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*.

Hansen, B. E. (2000). Sample splitting and threshold estimation. *Econometrica* **68** 575–603.

Hardt, M. (2014). Understanding alternating minimization for matrix completion. In *FOCS*. IEEE.

He, S., Yin, J., Li, H. and Wang, X. (2014). Graphical model selection and estimation for high dimensional tensor data. *Journal of Multivariate Analysis* **128** 165–185.

Jain, P., Netrapalli, P. and Sanghavi, S. (2013). Low-rank matrix completion using alternating minimization. In *STOC*.

Kalaitzis, A., Lafferty, J., Lawrence, N. and Zhou, S. (2013). The bigraphical lasso. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*.

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

Leng, C. and Tang, C. Y. (2012). Sparse matrix graphical models. *Journal of the American Statistical Association* **107** 1187–1200.

Mitchell, T. M., Hutchinson, R., Niculescu, R. S., Pereira, F., Wang, X., Just, M. and Newman, S. (2004). Learning to decode cognitive states from brain images. *Machine Learning* **57** 145–175.

Nesterov, Y. (2013). *Introductory lectures on convex optimization: A basic course*, vol. 87. Springer Science & Business Media.

Netrapalli, P., Jain, P. and Sanghavi, S. (2013). Phase retrieval using alternating minimization. In *NIPS*.

Ning, Y. and Liu, H. (2013). High-dimensional semiparametric bigraphical models. *Biometrika* **100** 655–670.

Park, D., Kyrillidis, A., Caramanis, C. and Sanghavi, S. (2016). Finding low-rank solutions to matrix problems, efficiently and provably. *arXiv preprint arXiv:1606.03168* .

Ravikumar, P., Wainwright, M. J., Raskutti, G., Yu, B. et al. (2011). High-dimensional covariance estimation by minimizing 1-penalized log-determinant divergence. *Electronic Journal of Statistics* **5** 935–980.

Rendle, S. and Schmidt-Thieme, L. (2010). Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM.

Rothman, A. J., Bickel, P. J., Levina, E., Zhu, J. et al. (2008). Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics* **2** 494–515.

Sun, W., Wang, Z., Liu, H. and Cheng, G. (2015). Non-convex statistical optimization for sparse tensor graphical model. In *Advances in Neural Information Processing Systems*.

Tsiligkaridis, T., Hero, A. O. and Zhou, S. (2013). On convergence of kronecker graphical lasso algorithms. *Signal Processing, IEEE Transactions on* **61** 1743–1755.

Vasilescu, M. A. O. and Terzopoulos, D. (2002). Multilinear analysis of image ensembles: Tensorfaces. In *Computer VisionECCV 2002*. Springer, 447–460.

Wang, L., Ren, X. and Gu, Q. (2016a). Precision matrix estimation in high dimensional gaussian graphical models with faster rates. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*.

Wang, L., Zhang, X. and Gu, Q. (2016b). A unified computational and statistical framework for nonconvex low-rank matrix estimation. *arXiv preprint arXiv:1610.05275* .

Wang, L., Zhang, X. and Gu, Q. (2017). A universal variance reduction-based catalyst for nonconvex low-rank matrix recovery. *arXiv preprint arXiv:1701.02301* .

Wang, Z., Gu, Q., Ning, Y. and Liu, H. (2014). High dimensional expectation-maximization algorithm: Statistical optimization and asymptotic normality. *arXiv preprint arXiv:1412.8729* .

Xiong, L., Chen, X., Huang, T.-K., Schneider, J. G. and Carbonell, J. G. (2010). Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, vol. 10. SIAM.

Yin, J. and Li, H. (2012). Model selection and estimation in the matrix normal graphical model. *Journal of multivariate analysis* **107** 119–140.

Yuan, X.-T. and Zhang, T. (2013). Truncated power method for sparse eigenvalue problems. *The Journal of Machine Learning Research* **14** 899–925.

Zhao, T., Liu, H., Roeder, K., Lafferty, J. and Wasserman, L. (2012). The huge package for high-dimensional undirected graph estimation in r. *The Journal of Machine Learning Research* **13** 1059–1062.

Zhao, T., Wang, Z. and Liu, H. (2015). A nonconvex optimization framework for low rank matrix estimation. In *Advances in Neural Information Processing Systems*.

Zheng, Q. and Lafferty, J. (2015). A convergent gradient descent algorithm for rank minimization and semidefinite programming from random linear measurements. In *Advances in Neural Information Processing Systems*.

Zhou, H., Li, L. and Zhu, H. (2013). Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association* **108** 540–552.

Zhou, S. et al. (2014). Gemini: Graph estimation with matrix variate normal instances. *The Annals of Statistics* **42** 532–562.