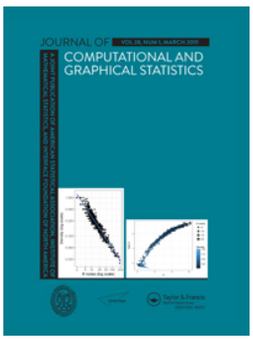


Taylor & Francis
Taylor & Francis Group

Journal of Computational and Graphical Statistics

ISSN: 1061-8600 (Print) 1537-2715 (Online) Journal homepage: https://www.tandfonline.com/loi/ucgs20

A Penalized Likelihood Method for Classification With Matrix-Valued Predictors

Aaron J. Molstad & Adam J. Rothman

To cite this article: Aaron J. Molstad & Adam J. Rothman (2019) A Penalized Likelihood Method for Classification With Matrix-Valued Predictors, Journal of Computational and Graphical Statistics, 28:1, 11-22, DOI: 10.1080/10618600.2018.1476249

To link to this article: https://doi.org/10.1080/10618600.2018.1476249

JOURNAL OF COMPUTATIONAL AND GRAPHICAL STATISTICS
2019, VOL. 28, NO. 1, 11–22
https://doi.org/10.1080/10618600.2018.1476249

Taylor & Francis
Taylor & Francis Group

A Penalized Likelihood Method for Classification With Matrix-Valued Predictors

Aaron J. Molstad[a] and Adam J. Rothman[b]

[a]Biostatistics Program, Fred Hutchinson Cancer Research Center, Seattle, WA; [b]School of Statistics, University of Minnesota, Minneapolis, MN

ABSTRACT

We propose a penalized likelihood method to fit the linear discriminant analysis model when the predictor is matrix valued. We simultaneously estimate the means and the precision matrix, which we assume has a Kronecker product decomposition. Our penalties encourage pairs of response category mean matrix estimators to have equal entries and also encourage zeros in the precision matrix estimator. To compute our estimators, we use a blockwise coordinate descent algorithm. To update the optimization variables corresponding to response category mean matrices, we use an alternating minimization algorithm that takes advantage of the Kronecker structure of the precision matrix. We show that our method can outperform relevant competitors in classification, even when our modeling assumptions are violated. We analyze three real datasets to demonstrate our method's applicability. Supplementary materials, including an R package implementing our method, are available online.

ARTICLE HISTORY
Received September 2016
Revised February 2018

KEYWORDS
Alternating minimization algorithm; Classification; Penalized likelihood

1. Introduction

Matrix-valued data are collected in many areas of scientific study. These include EEG experiments, where electrical activity on a subject's scalp is measured at c channels over r time points (Ingber 1998); longitudinal genomic studies, where subjects have the expression of c genes measured at r time points (Baranzini et al. 2004); and in imaging, where on each subject, some $r \times c$ pixel image is recorded. In these applications, fitted models are often used to classify a subject. For instance, in some EEG experiments, practitioners want to classify a subject as predisposed to alcoholism or not.

In this article, we propose a new method for classification when the predictor is matrix-valued. Although standard vector-valued predictor classification methods, such as logistic regression and linear discriminant analysis, could be applied, they would not take advantage of the matrix structure.

Logistic regression-based methods for classification with a matrix-valued predictor have been proposed. Zhou and Li (2014) proposed a nuclear norm penalized likelihood estimator of the regression coefficient matrix $B_* \in \mathbb{R}^{r\times c}$ in a generalized linear model, where the value of the matrix predictor $x \in \mathbb{R}^{r\times c}$ enters the model through the trace of $B_*^{\mathrm{T}}x$. In the same setup, Hung and Wang (2013) assumed that $\mathrm{vec}(B_*) = \beta_* \otimes \alpha_*$ where vec stacks the columns of its argument, $\alpha_* \in \mathbb{R}^r$, $\beta_* \in \mathbb{R}^c$, and $\otimes$ is the Kronecker product. This decomposition was also studied in the dimension reduction literature (Li, Kim, and Altman 2010).

There also exist methods that modify Fisher's linear discriminant criterion for matrix-valued predictors, for example, 2D-LDA (Li and Yuan 2005), matrix discriminant analysis (Zhong and Suslick 2015), and penalized matrix discriminant analysis (Zhong and Suslick 2015).

We propose a penalized likelihood method for classification with a matrix-valued predictor. Our method estimates the parameters in the linear discriminant analysis model. Let $x_i \in \mathbb{R}^{r\times c}$ be the measured predictor for the ith subject and let $y_i \in \{1, \ldots, J\}$ be the measured categorical response for the ith subject ($i = 1, \ldots, n$). We assume that $(x_1, y_1), \ldots, (x_n, y_n)$ are a realization of n independent copies of (X, Y) with the following distribution. The marginal distribution of Y is defined by $P(Y = j) = \pi_{*j}$ $(j = 1, \ldots, J)$, where the π_{*j}'s are unknown; and

$$\mathrm{vec}(X) \mid Y = j \sim \mathrm{N}_{rc}\left\{\mathrm{vec}\left(\mu_{*j}\right), \Sigma_*\right\}, \quad j = 1, \ldots, J, \quad (1)$$

where $\mu_{*j} \in \mathbb{R}^{r\times c}$ is the unknown mean matrix for the jth response category, and Σ_* is the unknown rc by rc covariance matrix.

When rc is large relative to the sample size n, the maximum likelihood estimator of Σ_*^{-1} may not exist, so shrinkage or parsimonious reparameterization is required. It is common to assume that Σ_* is diagonal (Bickel and Levina 2004) or has many entries which are zero (Xu et al. 2015), but without modification, these assumptions would not account for the matrix-structure of the predictor. Instead, we make the simplifying assumption that

$$\Sigma_*^{-1} = \Delta_* \otimes \Phi_*, \quad (2)$$

which is equivalent to $\Sigma_* = \Delta_*^{-1} \otimes \Phi_*^{-1}$, where Φ_* and Δ_* are unknown r by r and c by c precision matrices, respectively, with $\sum_{a,b}|\Phi_{*a,b}| = r$, where $\Phi_{*a,b}$ is the (a, b)th entry of Φ_*. The norm condition on Φ_* is required for identifiability: see Roś et al. (2016) for more on identifiability under (2).

This simplification of a covariance matrix makes the conditional distributions in (1) become matrix-normal (Gupta and Nagar 2000). In particular, the correlation matrices correspond-

CONTACT Aaron J. Molstad amolstad@fredhutch.org Biostatistics Program, Fred Hutchinson Cancer Research Center, Seattle, WA 98109.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JCGS.

ing to Φ_* and Δ_* can be interpreted in terms of entries in the rows and entries in the columns of the predictor, respectively (Hoff 2011). This exploits the matrix structure of the predictor by reducing the number of precision matrix parameters from $O(r^2c^2)$ to $O(r^2+c^2)$. Parameterizing the covariance of matrix-valued random variables using (2) is common in a wide range of applications involving matrix-valued data, for example, microarray data (Allen and Tibshirani 2010), longitudinal network data (Hoff 2011), financial data (Leng and Tang 2012), and neuroimaging data (Xia and Li 2017; Li and Zhang 2017).

Several authors have proposed and studied penalized likelihood estimators of Φ_* and Δ_* when $J=1$ (Allen and Tibshirani 2010; Zhang and Schneider 2010; Tsiligkaridis, Hero, and Zhou 2012; Leng and Tang 2012; Zhou 2014).

In this article, we propose a penalized likelihood method to fit (1) with the assumption in (2). Our penalties encourage fitted models that can be easily interpreted by practitioners. We use a blockwise coordinate descent algorithm to compute our estimators. To exploit (2) computationally, we use an alternating minimization algorithm (Tseng 1991) in one of our block updates. This algorithm scales more efficiently than other popular algorithms, which makes our method computationally feasible for high-dimensional problems. We show that our algorithm has the same computational complexity order as the unpenalized likelihood version, which also requires a blockwise coordinate descent algorithm (Dutilleul 1999).

2. Penalized Likelihood Estimation

2.1. Proposed Method

Let $\mathbb{S}_+^m$ be the set of symmetric and positive definite m by m matrices. The maximum likelihood estimators of the μ_{*j}'s, Φ_*, and Δ_* minimize the function $g:(\mathbb{R}^{r\times c})^J\times\mathbb{S}_+^r\times\mathbb{S}_+^c\to\mathbb{R}$ defined by

$$g(\mu,\Phi,\Delta)=\frac{1}{n}\sum_{j=1}^{J}\left[\sum_{i=1}^{n}1(y_i{=}j)\mathrm{tr}\left\{\Phi(x_i{-}\mu_j)\Delta(x_i{-}\mu_j)^{\mathrm{T}}\right\}\right]$$
$$-c\log\det(\Phi)-r\log\det(\Delta),$$

where $\mu=(\mu_1,\ldots,\mu_J)$. We propose the penalized likelihood estimators defined by

$$(\hat\mu,\hat\Phi,\hat\Delta)=\underset{(\mu,\Phi,\Delta)\in\mathcal{T}}{\arg\min}\left\{g(\mu,\Phi,\Delta)+\lambda_1\sum_{j<m}\|w_{j,m}\circ(\mu_j{-}\mu_m)\|_1\right.$$
$$\left.+\lambda_2\|\Delta\otimes\Phi\|_1\right\},\quad\text{subject to }\|\Phi\|_1=r\tag{3}$$

where $\mathcal{T}=(\mathbb{R}^{r\times c})^J\times\mathbb{S}_+^r\times\mathbb{S}_+^c$; $\circ$ is the Hadamard product; $\|\cdot\|_1$ is the sum of the absolute values of the entries of its argument; λ_1 and λ_2 are nonnegative tuning parameters; and the $w_{j,m}$'s are r by c user-specified weight matrices with nonnegative entries. We recommend selecting weights similar to those prescribed by Guo (2010). We suggest using

$$w_{j,m}^{-1}=|\bar x_j-\bar x_m|,\quad 1\le j<m\le J,$$

where $\bar x_j=n_j^{-1}\sum_{i=1}^n 1(y_i=j)x_i$ and $n_j=\sum_{i=1}^n 1(y_i=j)$. Alternatively, one could use weights based on t-test statistics or could use weights that incorporate prior information.

The first penalty in (3) encourages solutions for which pairs of the mean matrix estimates have some equal entries, where this equality occurs in the same locations. Without the first penalty, that is, $\lambda_1=0$, the proposed estimators of the μ_{*j}'s are sample mean matrices. If $\lambda_1>0$, then the proposed estimators of the μ_{*j}'s are affected by the estimators of Φ_* and Δ_*. In the supplementary material, we present numerical examples demonstrating that joint estimation of the μ_{*j}'s and (Φ_*,Δ_*) using (3) is superior to separate estimation.

The second penalty in (3) has a simple impact: for sufficiently large values of λ_2, some of the entries in the estimate of $\Delta_*\otimes\Phi_*$ are zero, which occurs if and only if either the estimate of Δ_* or the estimate of Φ_* has some zero entries. To encourage zeros in estimates of Φ_* or Δ_* separately, one could use two separate L_1 penalties. Our computational algorithm can be easily adapted to accommodate this case.

The tuning parameters λ_1 and λ_2 can be chosen by minimizing the misclassification rate on a validation set, or by maximizing a validation likelihood.

2.2. Related Work

Xu et al. (2015) proposed fitting the standard linear discriminant analysis model for a vector-valued predictor by penalized likelihood. We can express their parameter estimates in our matrix-predictor setup by setting the number of columns of the matrix predictor to one. Specifically, with $c=1$ and $\Delta=1$, Xu et al. (2015) parameter estimates are

$$\underset{(\mu,\Phi)\in(\mathbb{R}^r)^J\times\mathbb{S}_+^r}{\arg\min}\left\{g(\mu,\Phi,1)+\lambda_1\sum_{j<m}\|w_{j,m}\circ(\mu_j{-}\mu_m)\|_1\right.$$
$$\left.+\lambda_2\sum_{a\neq b}|\Phi_{ab}|\right\}.\tag{4}$$

One could view our method as the matrix-valued predictor extension of the method of Xu et al. (2015). Guo (2010) proposed a method that solves a restricted version of (4), where Φ is fixed at a diagonal matrix with pooled sample precision estimates on its diagonal.

Computationally, the algorithms proposed by Xu et al. (2015) and Guo (2010) for solving (4) suffer from numerical instability and do not scale efficiently for application to (3). In our simulation studies, we compare our proposed method to several competitors, including the method of Guo (2010). The method of Xu et al. (2015) is too slow computationally for the dimensions we consider, so we only use it in a special case when Σ_* is known.

Joint estimation of mean and precision matrix parameters using a penalized likelihood has also been proposed in the context of multivariate response linear regression (Rothman, Levina, and Zhu 2010; Yin and Li 2011; Lee and Liu 2012).

2.3. Statistical Theory

We expect that one could establish convergence rate bounds for (3) similar to those obtained by Leng and Tang (2012) for the precision matrix components and to those obtained by Xu et al. (2015) for the mean matrices by applying the same arguments used in the proof of Theorem 1 of Xu et al. (2015). However, because our objective function is nonconvex, these convergence

rate bounds are for some local minimizer, which may or may not be the local minimizer to which our algorithm converges.

3. Computation

3.1. Overview

To solve (3), we use a block-wise coordinate descent algorithm. Each block update is a convex optimization problem. In the subsequent subsections, we show that updates for Φ and Δ can be expressed as the well-studied L_1-penalized normal likelihood precision matrix estimation problem. We also use an alternating minimization algorithm for the block update for μ. The algorithm to compute our estimator, along with a set of auxiliary functions, is implemented in the R package `MatrixLDA`, which is available in the supplementary materials and on CRAN (R Core Team 2017).

3.2. Updates for Φ and Δ

We first derive the update for Φ. Define $\mathrm{GL}(S, \tau)$ as

$$\mathrm{GL}(S, \tau) = \underset{\Theta \in \mathbb{S}_+}{\arg\min} \left\{ \mathrm{tr}(S\Theta) - \log|\Theta| + \tau \|\Theta\|_1 \right\}, \tag{5}$$

where S is some given nonnegative definite matrix and τ is a nonnegative tuning parameter. The optimization problem in (5) is the L_1-penalized normal likelihood precision matrix estimation problem. Many algorithms and efficient implementations exist to solve (5): one good example is the `glasso` package in R (Friedman, Hastie, and Tibshirani 2008; Witten, Friedman, and Simon 2011), which we use in our implementation.

Let f be the objective function in (3). Suppose Δ and μ are fixed. The minimizer of f with respect to Φ is

$$\tilde{\Phi} = \underset{\Phi \in \mathbb{S}_+^r}{\arg\min} \left\{ \frac{1}{n} \sum_{j=1}^{J} \left[\sum_{i=1}^{n} 1(y_i = j) \mathrm{tr}\left\{ \Phi (x_i - \mu_j) \Delta (x_i - \mu_j)^{\mathrm{T}} \right\} \right] - c \log\det(\Phi) + \lambda_2 \|\Delta \otimes \Phi\|_1 \right\}. \tag{6}$$

Using the fact that $\|\Delta \otimes \Phi\|_1 = \|\Delta\|_1 \|\Phi\|_1$ and

$$\frac{1}{n} \sum_{j=1}^{J} \left[\sum_{i=1}^{n} 1(y_i = j) \mathrm{tr}\left\{ \Phi (x_i - \mu_j) \Delta (x_i - \mu_j)^{\mathrm{T}} \right\} \right] = c \ \mathrm{tr}\left\{ \Phi S_\phi(\mu, \Delta) \right\},$$

where

$$S_\phi(\mu, \Delta) = \frac{1}{nc} \sum_{j=1}^{J} \left\{ \sum_{i=1}^{n} 1(y_i = j)(x_i - \mu_j) \Delta (x_i - \mu_j)^{\mathrm{T}} \right\},$$

we can express (6) as

$$\underset{\Phi \in \mathbb{S}_+^r}{\arg\min} \left[\mathrm{tr}\left\{ \Phi S_\phi(\mu, \Delta) \right\} - \log|\Phi| + \frac{\lambda_2 \|\Delta\|_1}{c} \|\Phi\|_1 \right] = \mathrm{GL}\left\{ S_\phi(\mu, \Delta), \frac{\lambda_2 \|\Delta\|_1}{c} \right\}.$$

After computing $\tilde{\Phi}$ with Δ fixed, we can enforce the constraint $\|\Phi\|_1 = r$ using a simple normalization: we replace $(\tilde{\Phi}, \Delta)$ with $(\bar{\Phi}, \bar{\Delta})$, where

$$\bar{\Phi} = \frac{r}{\|\tilde{\Phi}\|_1} \tilde{\Phi}, \quad \bar{\Delta} = \frac{\|\tilde{\Phi}\|_1}{r} \Delta.$$

This ensures that $\|\bar{\Phi}\|_1 = r$ without changing the objective function because $f(\mu, \Delta, \tilde{\Phi}) = f(\mu, \bar{\Delta}, \bar{\Phi})$.

Using a similar argument, the minimizer of f with respect to Δ with μ and Φ fixed is

$$\tilde{\Delta} = \mathrm{GL}\left\{ S_\delta(\mu, \Phi), \frac{\lambda_2 \|\Phi\|_1}{r} \right\},$$

where

$$S_\delta(\mu, \Phi) = \frac{1}{nr} \sum_{j=1}^{J} \left\{ \sum_{i=1}^{n} 1(y_i = j) (x_i - \mu_j)^{\mathrm{T}} \Phi (x_i - \mu_j) \right\}.$$

3.3. Update for μ

Let Δ and Φ be fixed. The minimizer of f with respect to μ is

$$\underset{\mu \in \mathbb{R}^{(r \times c)J}}{\arg\min} \left\{ \frac{1}{n} \sum_{j=1}^{J} \left[\sum_{i=1}^{n} 1(y_i = j) \mathrm{tr}\{ \Phi (x_i - \mu_j) \Delta (x_i - \mu_j)^{\mathrm{T}} \} \right] + \lambda_1 \sum_{j<m} \| w_{j,m} \circ (\mu_j - \mu_m) \|_1 \right\}. \tag{7}$$

Special cases of (7) have been solved using an algorithm exploiting the majorize-minimize (MM) principle, where the penalty is majorized by its local-quadratic approximation at the current iterate (Hunter and Li 2005). For example, Xu et al. (2015) solved (7) when $c = 1$ and $\Delta = 1$; and Guo (2010) solved (7) when $c = 1$, $\Delta = 1$, and Φ was diagonal. However, this MM algorithm suffers from numerical instability when iterates for μ_j and μ_m are similar from some (j, m). Moreover, if we were to apply the MM algorithm to solve (7), then each iteration would require solving an $rc \times rc$ linear system of equations.

Instead of using an MM algorithm, we use an alternating minimization algorithm (Tseng 1991; Chi and Lange 2015) to solve (7). Our algorithm for solving (7) is more numerically stable, each iteration has worst case computational complexity $O(r^2c + c^2r)$, and has a quadratic rate of convergence when implemented with the accelerations proposed by Goldstein et al. (2014).

Similarly to the setup of the ADMM algorithm (Boyd et al. 2011), we first express (7) as a constrained optimization problem:

$$\begin{aligned} &\text{minimize}_{(\mu, \Theta) \in \mathcal{G}} \ \ g(\mu, \Phi, \Delta) + \lambda_1 \textstyle\sum_{j<m} \|w_{j,m} \circ \Theta_{j,m}\|_1 \\ &\text{subject to } \Theta_{j,m} = \mu_j - \mu_m \quad 1 \le j < m \le J, \end{aligned} \tag{8}$$

where $\mathcal{G} = \mathbb{R}^{(r \times c)J} \times \mathbb{R}^{(r \times c)\binom{J}{2}}$ and $\Theta = (\Theta_{1,2}, \ldots, \Theta_{J-1,J})$. The augmented Lagrangian for (8), using notation similar to Chi

and Lange (2015), is

$$\mathcal{F}_\rho(\mu, \Theta, \Gamma) = g(\mu, \Phi, \Delta) + \lambda_1 \sum_{j<m} \|w_{j,m} \circ \Theta_{j,m}\|_1 + \sum_{j<m} \operatorname{tr}\{\Gamma_{j,m}^{\mathrm{T}}(\Theta_{j,m} - \mu_j + \mu_m)\} + \frac{\rho}{2} \sum_{j<m} \|\Theta_{j,m} - \mu_j + \mu_m\|_F^2,$$

for step size parameter $\rho > 0$ and Lagrangian variables $\Gamma_{j,m} \in \mathbb{R}^{r\times c}$ for $1 \le j < m \le J$. Letting the superscript t denote the value of the tth iterate of an optimization variable, the alternating minimization algorithm updates

$$\mu^{(t+1)} \leftarrow \underset{\mu\in\mathbb{R}^{(r\times c)J}}{\arg\min}\, \mathcal{F}_0\left(\mu, \Theta^{(t)}, \Gamma^{(t)}\right), \tag{9}$$

$$\Theta^{(t+1)} \leftarrow \underset{\Theta\in\mathbb{R}^{(r\times c)\binom{J}{2}}}{\arg\min}\, \mathcal{F}_\rho\left(\mu^{(t+1)}, \Theta, \Gamma^{(t)}\right), \tag{10}$$

$$\Gamma_{j,m}^{(t+1)} \leftarrow \Gamma_{j,m}^{(t)} + \rho\left(\Theta_{j,m}^{(t+1)} - \mu_j^{(t+1)} + \mu_m^{(t+1)}\right) \text{ for } 1 \le j < m \le J, \tag{11}$$

until convergence. The ADMM algorithm modifies (9) by using $\mathcal{F}_\rho$ rather than $\mathcal{F}_0$. The advantage of using $\mathcal{F}_0$ is that we avoid solving an $rc \times rc$ linear system of equations (which can be solved with complexity $O(r^2c^2)$ when using the Kronecker structure). Using $\mathcal{F}_0$ also allows the updates for $\mu_1, \ldots, \mu_J$ to be computed in parallel with closed-form solutions for each, and removes the dependence on $\Theta^{(t)}$ from the update in (9). As a consequence, we can solve (10) and (11) simultaneously without computing or storing $\Theta^{(t+1)}$ explicitly, which we show later in this section.

Two conditions for the convergence of alternating minimization are that g is strongly convex (Tseng 1991), which it is in our case, and that ρ is sufficiently close to zero. We provide a computable bound on the step size ρ to ensure convergence of our alternating minimization algorithm in the subsequent section.

The computational advantage of alternating minimization over ADMM was also recognized by Chi and Lange (2015) in the context of convex clustering. They found that the simplification of (9) relative to the ADMM version yielded a substantially more efficient algorithm.

Using the first-order optimality condition for (9),

$$\mu_j^{(t+1)} = \bar{x}_j + \frac{1}{2\hat{\pi}_j}\Phi^{-1}\left(\sum_{\{m:m>j\}} \Gamma_{j,m}^{(t)} - \sum_{\{m:m<j\}} \Gamma_{m,j}^{(t)}\right)\Delta^{-1} \qquad j = 1, \ldots, J, \tag{12}$$

where $\hat{\pi}_j = n_j/n$ for $j = 1, \ldots, J$.

Plugging the right-hand side of (10) into the updating Equation (11) and applying the Moreau decomposition (e.g., Section 2.5 of Parikh and Boyd 2014), it follows that (10) and (11) can be solved simultaneously by setting

$$\Gamma_{j,m}^{(t+1)} = \mathcal{C}\big(\Gamma_{j,m}^{(t)} - \rho\mu_j^{(t+1)} + \rho\mu_m^{(t+1)}, \lambda_1 w_{j,m}\big), \tag{13}$$

where $\mathcal{C} : \mathbb{R}^{a\times b} \times \mathbb{R}^{a\times b} \to \mathbb{R}^{a\times b}$ is the function defined by

$$[\mathcal{C}(\Omega, M)]_{s,t} = \begin{cases} |M_{s,t}| : \Omega_{s,t} \ge |M_{s,t}| \\ -|M_{s,t}| : \Omega_{s,t} \le -|M_{s,t}| \\ \Omega_{s,t} : \text{otherwise} \end{cases}, (s,t) \in [1, \ldots, a] \times [1, \ldots, b].$$

Thus, we need not compute or store the iterates of Θ, which saves $O(rc)$ in memory complexity relative to the ADMM algorithm that solves (7). This simplification was also derived in Chi and Lange (2015) and in Zhu (2017).

To summarize, the alternating minimization algorithm solves (9) using (12) and solves (10) and (11) simultaneously using (13). In our implementation, we use an accelerated variation of the algorithm presented in this section to solve (7). Further details about our implementation are given in the subsequent section.

3.4. Summary

The block-wise coordinate descent algorithm for solving (3) is presented in Algorithm 1.

Algorithm 1. Given $\epsilon > 0$, $\Delta^{(0)} \in \mathbb{S}_+^c$, $\Phi^{(0)} \in \mathbb{S}_+^r$ such that $\|\Phi^{(0)}\|_1 = r$. Set $k = 0$:

- Step 1: Compute $\mu^{(k+1)} = \underset{\mu\in\mathbb{R}^{(r\times c)J}}{\arg\min}\, g(\mu, \Phi^{(k)}, \Delta^{(k)}) + \lambda_1 \sum_{j<m} \|w_{j,m} \circ (\mu_j - \mu_m)\|_1$ using the algorithm from Section 3.3
- Step 2: Compute $\tilde{\Delta} = \mathrm{GL}\{S_\delta(\mu^{(k+1)}, \Phi^{(k)}), \lambda_2\}$.
- Step 3: Compute $\tilde{\Phi} = \mathrm{GL}\{S_\phi(\mu^{(k+1)}, \tilde{\Delta}), \frac{\lambda_2}{c}\|\tilde{\Delta}\|_1\}$.
- Step 4: Compute $\Delta^{(k+1)} = \frac{\|\tilde{\Phi}\|_1}{r}\tilde{\Delta}$, $\Phi^{(k+1)} = \frac{r}{\|\tilde{\Phi}\|_1}\tilde{\Phi}$.
- Step 5: If $f(\mu^{(k)}, \Phi^{(k)}, \Delta^{(k)}) - f(\mu^{(k+1)}, \Phi^{(k+1)}, \Delta^{(k+1)}) < \epsilon|f(\bar{x}, \Phi^{(0)}, \Delta^{(0)})|$, then stop. Otherwise, replace k by $k+1$ and go to step 1.

In our implementation, we set $\epsilon = 10^{-6}$. To get initial values $\Phi^{(0)}$ and $\Delta^{(0)}$, we run the maximum likelihood algorithm (Dutilleul 1999) until a mild convergence tolerance is reached, and use $\Phi^{(0)} = \mathrm{diag}(\Phi^{\mathrm{MLE}})$ and $\Delta^{(0)} = \mathrm{diag}(\Delta^{\mathrm{MLE}})$ where $(\Phi^{\mathrm{MLE}}, \Delta^{\mathrm{MLE}})$ are the final iterates.

Let $\kappa_\phi^{(k)} = \varphi_{\min}(\Phi^{(k)})$ and $\kappa_\delta^{(k)} = \varphi_{\min}(\Delta^{(k)})$, where $\varphi_{\min}$ denotes the minimum eigenvalue. For the $(k+1)$th update of μ, if we select the step size parameter $\rho^{(k+1)} \in (0, 4\min_j\{\hat{\pi}_j\}\kappa_\phi^{(k)}\kappa_\delta^{(k)}/J)$, then the alternating minimization algorithm converges (Tseng 1991; Chi and Lange 2015). One can verify that (8) and $\rho^{(k+1)}$ satisfy the conditions for convergence stated in sec. 6.2 of the supplemental material of Chi and Lange (2015) using an argument similar to theirs. The minimum eigenvalues of $\Phi^{(k)}$ and $\Delta^{(k)}$ are positive as long as initializers $\Phi^{(0)}$ and $\Delta^{(0)}$ are positive definite. When $\kappa_\delta^{(k)}$ and $\kappa_\phi^{(k)}$ are positive, g is strongly convex in μ, which is required for convergence. In practice, we use ρ order of magnitude smaller than the required upper bound, that is, we use $\rho^{(k+1)} = 4\min_j\{\hat{\pi}_j\}\kappa_\phi^{(k)}\kappa_\delta^{(k)}/(10J)$.

We use an accelerated version of the alternating minimization algorithm proposed by Goldstein et al. (2014), which was also used by Chi and Lange (2015). We warm-start the $(k+1)$th

update of μ by initializing the Lagrangian variables at their final iterates from the kth update.

3.5. Computational Properties

Solving (3) with $\lambda_1 = \lambda_2 = 0$, that is, maximum likelihood estimation, also requires a blockwise coordinate descent algorithm (Dutilleul 1999). The maximum-likelihood blockwise coordinate descent algorithm has computational complexity $O(nr^2c + nc^2r + r^3 + c^3)$. The first two terms come from computing the sample covariance matrices S_ϕ and S_δ, and the last two terms come from inverting S_ϕ and S_δ.

Our algorithm's computational complexity is also $O(nr^2c + nc^2r + r^3 + c^3)$. We compute S_ϕ and S_δ and the graphical-lasso algorithm that we use is known to have worst case complexity $O(p^3)$ for estimating a $p \times p$ precision matrix (Friedman, Hastie, and Tibshirani 2008). In addition, for each μ update, we compute eigendecompositions of the iterates for Φ and Δ which are used to compute $\rho^{(k+1)}$ and solve (12).

Because each step of our algorithm solves a convex optimization problem, we are guaranteed to decrease the objective function at each step. However, the objective function is nonconvex so we are unable to say anything about whether our iterates reach a global minimizer. Nonetheless, in the settings we considered, our final iterates satisfied the first-order optimality conditions.

4. Simulation Study

4.1. Models

For 100 independent replications, we generated a realization of $n = n_{\text{train}} + n_{\text{validate}} + n_{\text{test}}$ independent copies of (X, Y), where we set $n_{\text{train}} = n_{\text{validate}} = 75$, and $n_{\text{test}} = 1000$. The categorical response Y has support $\{1, 2, 3\}$ with probabilities $\pi_{*1} = \pi_{*2} = \pi_{*3} = 1/3$. Then

$$\text{vec}\,(X) \mid Y = j \sim \text{N}_{rc}\left\{\text{vec}\left(\mu_{*j}\right), \Sigma_*\right\},$$

where μ_{*1}, μ_{*2}, and μ_{*3} are only different in one 4×4 submatrix, whose position is chosen randomly in each replication. We used multiple choices for the entries in this submatrix, which are displayed in Figure 1. All other mean matrix entries were set to zero. We considered four covariance models:

- Model 1. $\Sigma_* = \Delta_* \otimes \Phi_*$, where Φ_* has (a, b)th entry $0.5^{|a-b|}$ and Δ_* has (c, d)th entry $0.5 \times 1(c \neq d) + 1(c = d)$.
- Model 2. $\Sigma_* = \Delta_* \otimes \Phi_*$, where Φ_* has (a, b)th entry $0.5^{|a-b|}$ and Δ_* is block-diagonal where Δ_* can be expressed elementwise:

$$\Delta_{c,d} = \begin{cases} 1 & \text{if } c = d \\ 0.8 & \mu_{*l,a,c} \neq \mu_{*m,a,d} \text{ and } c \neq d \\ 0 & \text{otherwise} \end{cases}.$$

  for any $a \in \{1, \ldots, r\}$ and $1 \leq l < m \leq J$
- Model 3. Σ_* corresponds to the covariance model

$$\text{cov}(X_{a,b}, X_{c,d} \mid Y = j) = \{0.5I(b \neq d) + I(b = d)\} \frac{(\rho_b \rho_d)^{|a-c|}}{1 - \rho_b \rho_d},$$

  where $\rho_1, \ldots, \rho_c$ are c equally spaced values between 0.5 and 0.9. The matrix Σ_* is positive definite when $r = c$ with $c = \{8, 16, 32, 64\}$, and when $r = 32$ with $c = \{8, 16, 32, 64\}$.
- Model 4. Σ_* corresponds to the covariance model

$$\text{cov}(X_{a,b}, X_{c,d} \mid Y = j) = \begin{cases} 1 & \text{if } (a, b) = (c, d) \\ 0.5 & \text{if } \mu_{*l,a,b} \neq \mu_{*m,c,d} \\ 0 & \text{otherwise} \end{cases}.$$

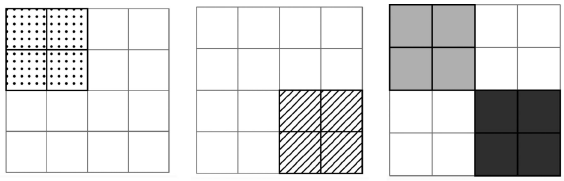
  for any $1 \leq l < m \leq J$

In Model 3, if $\rho_k = \rho$ for all $k \in \{1, \ldots, c\}$, then Σ_* has the decomposition (2) corresponding to Φ_* with an AR(1) structure and Δ_* with a compound symmetric structure (Mitchell, Genton, and Gumpertz 2006). However, when $\rho_k \neq \rho$, Σ_* does not have the decomposition (2): the covariance between any two entries in the same row depends on the column and vice versa. Model 4 is the $rc-$variate normal model similar to the first model used in the simulations from Xu et al. (2015).

4.2. Methods

We consider the following model-based methods for fitting the linear discriminant analysis model:

- Bayes. The Bayes rule, that is, Σ_*, μ_*, and π_{*j} known for $j = 1, \ldots, J$;
- MN. The maximum likelihood estimator of (1) under (2), that is, the matrix-normal maximum likelihood estimator;
- Guo. The sparse naïve Bayes type estimator proposed by Guo (2010) defined in Section 2.2 with tuning parameter chosen to minimize the misclassification rate on the validation set;
- vec-SURE. The multiclass SURE independence screening method proposed by Pan, Wang, and Li (2016) with model sizes chosen to minimize the misclassification rate on the validation set;
- MN-SURE. A matrix-normal extension of the SURE independence screening estimator proposed by Pan, Wang, and



(a) μ_{*1} (b) μ_{*2} (c) μ_{*3}

Model				
1 & 2	1	.5	-2	-.5
3	1	-1.5	-2	-.5
4	1	1.5	-2.5	-1

Figure 1. The 4×4 submatrices where μ_{*1}, μ_{*2}, and μ_{*3} differ. White corresponds to zero and the legend gives the values corresponding to the highlighted cells for each model.

Li (2016) with model sizes chosen to minimize the misclassification rate on the validation set;
- PMN(μ). The estimator defined by (3) with $\mu = \mu_*$ fixed and λ_2 chosen to minimize the misclassification rate on the validation set;
- PMN(Σ)/Xu(Σ). The estimator defined by (3) with $\Phi = \Phi_*$ and $\Delta = \Delta_*$ fixed when $\Sigma_* = \Delta_* \otimes \Phi_*$; the estimator of Xu et al. (2015) with $\Sigma = \Sigma_*$ fixed when $\Sigma_* \neq \Delta_* \otimes \Phi_*$; and λ_1 chosen to minimize the misclassification rate on the validation set;
- PMN. The estimator defined by (3) with tuning parameters chosen to minimize the misclassification rate on the validation set.

The methods PMN(μ) and PMN(Σ)/Xu(Σ) both use some oracle information and were included to study how estimating μ_*, Δ_*, and Φ_* simultaneously affect classification accuracy. We refer to these methods as part-oracle matrix-LDA methods. We refer to Guo and vec-SURE as vector-LDA methods; MN and MN-SURE as nonoracle matrix-LDA methods. MN-SURE is a matrix-normal generalization of the screening method proposed by Pan, Wang, and Li (2016). We compare to these methods because they all fit the model in (1). In the supplementary material, we also compare a subset of these methods to vector-valued linear discriminant methods that do not fit (1).

Following Guo (2010), we use a validation set to select tuning parameters. The candidate set for tuning parameters was $\{2^x : x = -12, -11.5, \ldots, 11.5, 12\}$. Candidate model sizes for vec-SURE and MN-SURE were $\{0, 1, \ldots, 25\}$, where model size refers to the number of pairwise nonzero mean differences based on thresholding.

4.3. Performance Measures

To compare classification accuracy, we record the misclassification rate on the test set for each replication. We also measure identification of mean differences that are zero through both true positive rate (TPR) and true negative rate (TNR). Let $D(\mu_*) = [\text{vec}(\mu_{*1} - \mu_{*2}), \ldots, \text{vec}(\mu_{*(J-1)} - \mu_{*J})]$, and $D(\hat{\mu}) = [\text{vec}(\hat{\mu}_1 - \hat{\mu}_2), \ldots, \text{vec}(\hat{\mu}_{(J-1)} - \hat{\mu}_J)]$. We define TPR as

$$\text{TPR}(\hat{\mu}, \mu_*) = \frac{\#\left\{(z, w) : \left[D(\hat{\mu})\right]_{z,w} \neq 0 \cap [D(\mu_*)]_{z,w} \neq 0\right\}}{\#\left\{(z, w) : [D(\mu_*)]_{z,w} \neq 0\right\}},$$

where # denotes cardinality. We similarly define TNR as

$$\text{TNR}(\hat{\mu}, \mu_*) = \frac{\#\left\{(z, w) : \left[D(\hat{\mu})\right]_{z,w} = 0 \cap [D(\mu_*)]_{z,w} = 0\right\}}{\#\left\{(z, w) : [D(\mu_*)]_{z,w} = 0\right\}}.$$

TNR and TPR together address mean difference estimation which we use for comparison to the estimators of Guo (2010), Xu et al. (2015), Pan, Wang, and Li (2016).

4.4. Results

We display average misclassification rates for Models 1 and 2 in Figure 2. For Model 1, the matrix-normal maximum likelihood

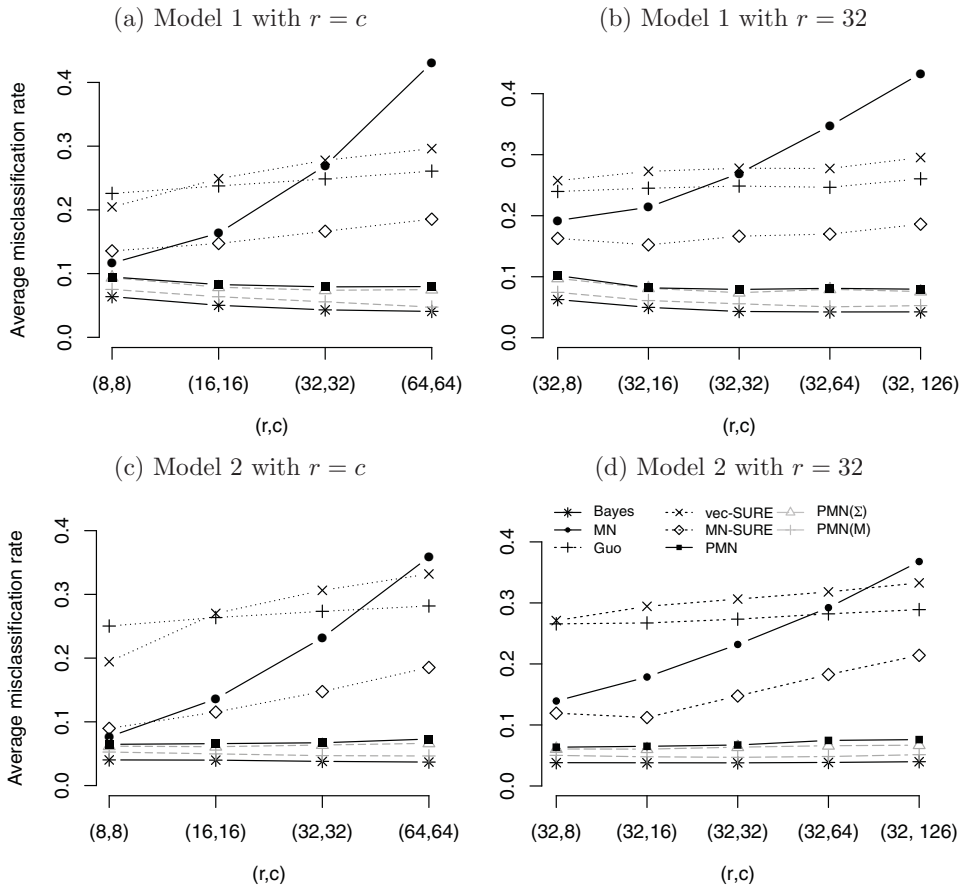


Figure 2. Misclassification rates averaged over 100 replications; (a) and (b) are for Model 1 and (c) and (d) for Model 2.

Table 1. TNR/TPR percentages averaged over the 100 replications for Models 1–4.

	Model 1 (r, c)							
Method	(8,8)	(16,16)	(32,32)	(64,64)	(32,8)	(32,16)	(32,64)	(32,126)
Guo	87.3/79.0	94.9/ 72.8	98.2/67.8	99.4/61.1	96.9/68.0	97.0/68.1	99.4/62.5	99.5/59.0
vec-SURE	86.6/71.8	98.0/53.4	99.5/39.9	99.9/31.1	98.5/47.0	99.0/46.0	99.7/39.0	99.9/28.7
MN-SURE	43.1/86.2	80.9/70.4	95.3/48.4	98.6/37.9	85.8/65.6	88.4/63.5	97.4/48.4	98.9/38.1
PMN(Σ)	86.8/87.4	93.3/87.9	98.8/83.8	99.6/81.6	95.1/85.5	97.0/85.4	99.0/77.8	99.6/76.8
PMN	89.4/81.4	96.3/81.0	98.9/75.8	99.5/73.0	98.2/73.4	98.3/76.5	99.4/73.4	99.7/70.9
	Model 2 (r, c)							
	(8,8)	(16,16)	(32,32)	(64,64)	(32,8)	(32,16)	(32,64)	(32,126)
Guo	83.2/79.9	92.3/76.1	96.7/69.6	98.5/65.5	95.7/68.4	95.1/72.0	97.3/67.9	98.5/64.6
vec-SURE	84.6/69.5	98.2/50.9	99.5/39.6	99.9/27.4	98.1/47.4	99.1/45.0	99.8/33.9	99.9/30.2
MN-SURE	46.3/85.8	75.6/73.2	89.6/53.5	97.4/37.8	82.1/63.2	85.7/62.3	95.5/45.2	97.9/37.9
PMN(Σ)	87.7/88.4	95.1/88.2	98.5/86.0	99.6/85.5	94.3/87.1	97.7/84.8	99.3/85.1	99.7/85.4
PMN	92.2/78.5	97.3/81.8	99.3/74.0	99.8/70.9	98.4/74.5	98.7/75.4	99.5/73.1	99.7/70.4
	Model 3 (r, c)							
	(8,8)	(16,16)	(32,32)	(64,64)	(32,8)	(32,16)	(32,64)	(32,126)
Guo	86.8/84.2	95.5/81.2	97.9/75.1	99.4/61.6	95.3/80.4	96.0/79.8	99.4/61.6	— / —
vec-SURE	85.8/84.8	98.3/63.2	99.6/47.1	99.9/36.0	98.1/57.9	99.1/51.2	99.9/36.0	— / —
MN-SURE	49.7/92.8	83.9/84.0	95.3/66.6	98.9/49.4	87.3/78.0	90.2/74.8	98.9/49.4	— / —
Xu(Σ)	85.7/95.8	94.6/94.6	98.7/87.5	99.7/70.8	93.1/93.0	97.0/90.9	99.7/70.8	— / —
PMN	86.5/96.1	93.9/96.0	98.4/92.6	99.2/87.1	93.8/93.5	95.8/94.0	99.2/87.1	— / —
	Model 4 (r, c)							
	(8,8)	(16,16)	(32,32)	(64,64)	(32,8)	(32,16)	(32,64)	(32,126)
Guo	82.2/98.5	93.9/98.5	96.9/97.1	98.8/94.6	90.4/98.2	95.0/97.9	98.0/95.4	98.8/95.2
vec-SURE	97.9/81.6	99.4/80.0	99.8/77.0	99.9/68.4	99.2/80.5	99.5/79.8	99.9/73.9	99.9/71.4
MN-SURE	68.1/96.8	90.0/93.8	97.1/85.0	98.6/76.1	89.2/94.0	94.7/90.9	97.6/84.2	98.9/77.9
Xu(Σ)	89.9/96.0	97.4/96.1	99.3/94.5	99.9/92.2	96.9/94.5	98.9/94.1	99.8/91.8	99.9/92.9
PMN	92.6/96.6	95.8/97.5	97.3/94.9	99.6/90.6	94.3/96.4	97.6/95.5	99.0/91.2	99.2/93.5

estimator tended to outperform the vector-LDA methods when r and c were small, but its average classification rate got worse as the dimensionality increased. The estimator proposed by Guo (2010) performs poorly when r and c are small, but got worse more gradually than the other vector and nonoracle matrix-LDA methods. The misclassification rate of the Bayes rules suggests that as the dimensionality increases in Model 1, the optimal misclassification rate can be improved. Our method PMN had improved classification accuracy as both r and c increased and performed similarly to PMN(Σ), which uses some oracle information.

TPR and TNR results are displayed in Table 1. For Model 1, PMN tended to have the second highest TNR behind vec-SURE, but tends to have higher TPR than all competing methods except PMN(Σ), which uses some oracle information.

We also report computing time statistics for PMN for Model 1 in Table 2. In our simulations, we used a 12×12 grid of candidate tuning parameters (λ_1, λ_2) for our method. For each candidate pair of tuning parameters, we averaged their computing time over the 100 replications and report the sample statistics of these averages over the 144 candidate pairs. We also display statistics for the average computing times (without warm-start initialization) for the tuning parameter pair selected by minimizing the misclassification rate on the validation set.

Misclassification results were similar for Model 2. The matrix-normal variation of the SURE screening estimator of Pan, Wang, and Li (2016) tended to perform best among the vector and nonoracle matrix-LDA methods. The estimator of Guo (2010) got worse most gradually among the vector-LDA methods. PMN performed nearly as well as PMN(Σ), both of which performed more closely to PMN(μ) and the Bayes rule than for Model 1.

The misclassification rates for Models 3 and 4 are displayed in Figure 3. In Model 3, although Σ_* does not have the Kronecker decomposition in (2), PMN outperformed all nonoracle estimators. In terms of TPR and TNR results presented in Table 1, PMN performed similarly to Xu(Σ), both of which had higher TPR than competitors and TNR similar to vec-SURE. This suggests that even when (2) does not hold, our method can perform well in classification.

Table 2. Summary statistics for average computing time (in seconds) over 100 replications for PMN for Model 1. Candidate grid columns are the minimum, median, mean, and maximum average computing time for tuning parameter pairs from a 12×12 grid of candidate tuning parameters using warm-start initialization. The $(\hat{\lambda}_1, \hat{\lambda}_2)$ columns are the average computing time without warm-start initialization for the tuning parameter pair chosen to minimize the misclassification rate on the validation set.

	Average computing time (secs)						Average computing time (secs)					
	$r = c$						$r = 32$					
	Candidate grid				$(\hat{\lambda}_1, \hat{\lambda}_2)$		Candidate grid				$(\hat{\lambda}_1, \hat{\lambda}_2)$	
	Min	Median	Mean	Max	Median	Mean	Min	Median	Mean	Max	Median	Mean
$c = 8$	0.014	0.021	0.022	0.045	0.034	0.037	0.029	0.055	0.057	0.124	0.082	0.084
$c = 16$	0.027	0.059	0.059	0.103	0.090	0.089	0.043	0.111	0.111	0.221	0.162	0.160
$c = 32$	0.148	0.360	0.357	0.587	0.490	0.479	0.148	0.360	0.357	0.587	0.490	0.479
$c = 64$	1.936	4.360	4.404	8.631	5.389	5.271	1.452	3.006	2.968	4.309	3.597	3.508
$c = 126$	—	—	—	—	—	—	9.545	21.373	20.694	29.668	24.259	23.285

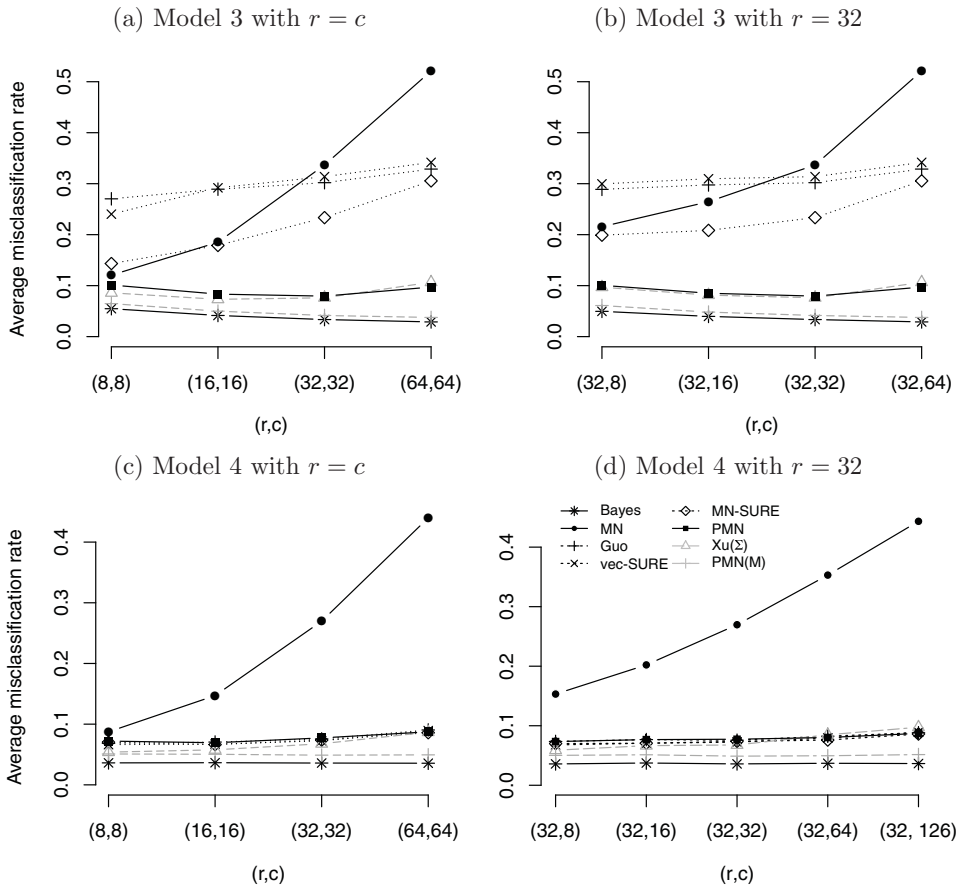


Figure 3. Misclassification rates averaged over 100 replications; (a) and (b) are for Model 3 and (c) and (d) for Model 4.

In Model 4, PMN performed similarly to the vector-LDA methods. MN-SURE was the best nonoracle method, which suggests that (2) may be a reasonable alternative to naïve Bayes under high dimensionality. Like in Model 3, PMN performed similarly to Xu(Σ) in terms of TPR and TNR.

In the supplementary material, we also compare our method to the vector-valued linear discriminant methods proposed by Witten and Tibshirani (2011), Clemmensen et al. (2011), and Mai, Yang, and Zou (2018) for Model 1. We found that our method performed significantly better than these vector-valued methods. We also compared our method to the matrix discriminant analysis methods proposed by Zhong and Suslick (2015). Using the exact data-generating model that they used in their simulations, our method performed only slightly worse than the penalized version of their method, but better than all other competitors they considered. When we modified their data-generating model to have a slightly stronger signal, that is, a lower Bayes misclassification rate, our method performed better than their methods and all competitors they considered.

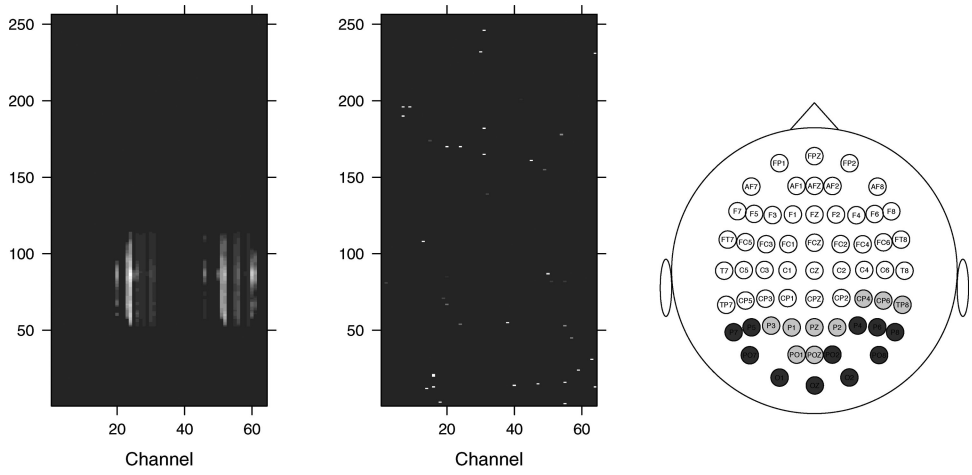
5. Data Examples

5.1. EEG Alcoholics Data

We analyzed the EEG data (*http://kdd.ics.uci.edu/databases/eeg/eeg.html*) also studied by Li, Kim, and Altman (2010) and Zhou and Li (2014). In the original study, 122 subjects, 77 of whom were alcoholics and 45 of whom were control, were exposed to stimuli while voltage was measured from $c = 64$ channels on a subject's scalp at $r = 256$ time points. Each subject underwent 120 trials. Each trial had one of three possible stimuli: single stimulus, two matched stimuli, or two unmatched stimuli. As in Li, Kim, and Altman (2010) and Zhou and Li (2014), we only analyzed the single stimulus condition. Because each subject underwent multiple trials under the single stimulus condition, we used the within subject average over all single stimulus trials as the predictor and we used whether they were alcoholic or control as the response.

It is common to assume that (2) holds in the analysis of EEG data. For example, Zhou (2014) assumed that (2) holds when analyzing a single subject from this same dataset. It may also be reasonable to assume that only a subset of channels and time point combinations are important for discriminating between alcoholic and control response categories. Thus, the primary goal of our analysis was to identify a subset of channels and time point combinations that help explain how the alcoholics and controls react to the stimulus differently.

To demonstrate our method's classification accuracy, we used the leave-one-out cross-validation approach from Li, Kim, and Altman (2010) and Zhou and Li (2014). For $k = 1, \ldots, 122$, we left out the kth observation and used the remaining 121 observations as training data. For each k, we selected tuning parameters for use in (3) by minimizing five-fold cross-validation misclassification rate on the training dataset. Our method correctly classified 97 of 122 subjects. Li, Kim, and Altman (2010) and Zhou and Li (2014) reported correctly classifying 97 and 94 of 122



(a) Proposal (b) Lasso (c) EEG Cap

Figure 4. (a) The absolute value of the estimated discriminant function using (3) with a tuning parameter pair that had leave-one-out classification accuracy 96/122. The darkest entries are equal to zero and lighter entries indicate a larger magnitude. (b) The absolute value of the logistic-regression coefficients estimated with vector-valued L_1-penalized logistic regression model using the tuning parameter chosen to minimize the misclassification rate using five-fold cross-validation. (c) An EEG cap, created using the `eegkit` package in R (Helwig 2015), with the channels colored based on the estimate in (a). Dark gray channels had 50 or more informative time points. Light gray channels had less than 50 but more than one informative time points.

subjects, respectively. Li, Kim, and Altman (2010) used quadratic discriminant analysis after dimension-folding of the predictors, and Zhou and Li (2014) used logistic regression with spectral regularization of the coefficient matrix.

While the method proposed by Li, Kim, and Altman (2010) performed similar to ours in terms of leave-one-out classification accuracy, interpreting their fitted model may be more difficult. Because their method uses dimension reduction, their fitted model must be interpreted in terms of linear combinations of all c channels and linear combinations of all r time points. In contrast, because our method exploits sparsity and scales to the dimensions of these data, our fitted model can be interpreted in terms of the original channels and time points.

To demonstrate the interpretability of our fitted models, we separately fit (3) using the complete dataset. We used a tuning parameter pair $(\lambda_1, \lambda_2) = (0.25, 8)$, which balanced leave-one-out classification accuracy (96/122) with model parsimony. Under the assumption (2), it can be shown that a zero in the (i, j)th entry of $\Phi_*(\mu_{*l} - \mu_{*m})\Delta_*$ implies that the (i, j)th component of the predictor matrix is noninformative for discriminating between response categories l and m (Xu et al. 2015). In Figure 4(a), we display the absolute values of the entries of $\hat{\Phi}(\hat{\mu}_A - \hat{\mu}_C)\hat{\Delta}$ estimated using (3), where $\hat{\mu}_A$ and $\hat{\mu}_C$ are the estimated mean matrices of the alcoholic and control response categories, respectively. Using our method, $\hat{\Phi}(\hat{\mu}_A - \hat{\mu}_C)\hat{\Delta}$ was only 5.5% nonzero. Following Zhou and Li (2014), we also display the absolute value of the estimated regression coefficients using vector-valued L_1-penalized logistic regression. While the L_1-penalized logistic regression coefficients, shown in Figure 4(b), selected a smaller number of informative channel and time point combinations, our method had better classification accuracy and selected channels neatly arranged in space and time. In Figure 4(c), we highlight the channels selected by our method as informative for discriminating between alcoholics and controls on an EEG cap. In particular, we estimated that 44 of the 64 channels were noninformative for discriminating between the two response categories. We estimated that no channels were informative before the 53rd or after the 114th time point. Interestingly, although our method does not explicitly use the spatial structure of channels or the temporal structure of the time points in estimation, we identified a set of important channels and time points which have a natural arrangement in space and time.

5.2. Longitudinal Genomic Data

We also analyzed the longitudinal genomic data from Baranzini et al. (2004). In the original study, researchers treated 53 multiple sclerosis patients with recombinant human interferon beta therapy. Patients had $c = 76$ genes' expression measured at 0, 3, 6, 9, 12, 18, and 24 months after therapy began. After monitoring ended, each patient was categorized as having had a positive or negative response to the therapy.

Because some of the data were missing, we omitted the 24 month time point and used only the first $r = 6$ time points. For all other missing entries, we imputed their values using the `mice` package in R (Buuren and Groothuis-Oudshoorn 2010). An R script to create the dataset we analyzed is available in the supplementary material.

Using our method with tuning parameters selected by five-fold cross-validation, we correctly classified 45 out of 53 subjects. For the competing methods, we report leave-one-out classification accuracy rates in Table 3. We found that our method performed best among the competitors we considered, including both matrix discriminant analysis methods proposed by Zhong and Suslick (2015). The unpenalized version of their method correctly classified 40 of 53 subjects. Unfortunately, due to the small sample size, we had difficulty using cross-validation to select tuning parameters for the penalized version of their method. On the complete dataset, the best leave-one-out classification rate the penalized version of their method achieved was 33/53 for the tuning parameters and initializing values we tried. For comparison, our method's best tuning parameter pair correctly classified 49 of 53 subjects.

Table 3. Leave-one-out classification rates on three datasets we analyzed using: PMN, our proposed estimator in (3); Lasso, the vector-valued L_1-penalized generalized linear model; Witten, the vector-valued L_1-penalized Fisher-linear-discriminant method proposed by Witten and Tibshirani (2011); and alternative methods applicable to each specific dataset.

Dataset	PMN	Lasso	Witten	Alternatives
EEG alcoholics	79.5%	75.4%	74.6%	79.5%[a], 77.0%[b]
Longitudinal genomic	84.9%	79.2%	73.5%	62.2%[c], 75.4%[d]
Hand gestures	90.0%	83.8%	65.0%	78.8%[e], 90.0%[f]

NOTES: (a) Dimension folding, reported in Li, Kim, and Altman (2010); (b) regularized matrix regression, reported in Zhou and Li (2014); (c) penalized matrix discriminant analysis (Zhong and Suslick 2015) with tuning parameter chosen to minimize the leave-one-out misclassification rate on the complete dataset; (d) matrix discriminant analysis (Zhong and Suslick 2015); (e) multiclass sparse linear discriminant analysis (Mai, Yang, and Zou 2018) with tuning parameter selected to minimize five-fold cross-validation misclassification rate; (f) neural networks.

Our fitted models can provide insight about differences between subjects who responded positive and negative to treatment. We fit (3) using the complete dataset with a tuning parameter pair that balanced leave-one-out classification accuracy (47/53) and model parsimony. Our fitted model estimated 11 of the 76 genes to have nonzero mean matrix differences for all six time points. Our estimated mean differences show no clear change over time. We also estimated the precision matrix corresponding to the time points to be diagonal. Together, these parameter estimates suggest that baseline gene expression may be associated with the response. A similar conclusion was reached in Baranzini et al. (2004) and Lyu, Lock, and Eberly (2017), who found no evidence of a gene-time interaction when analyzing different versions of this dataset.

Additionally, we can use the estimated precision components to study the partial correlations between the genes considered in this study. We display the estimated mean matrix differences and the precision matrix estimate corresponding to genes in the supplementary material.

5.3. Cambridge Hand Gestures Data

We used our method to classify eighty images taken from the Cambridge hand gestures database (Kim and Cipolla 2009). In the dataset we analyzed, each image depicts a hand in one of two shapes (flat or V) and one of two orientations (centered or left). The original 80 color images (20 of each shape and orientation combination) were transformed to grayscale and resized from 320×240 to $r \times c = 80 \times 60$ pixels using the `EBImage` package in R (Pau et al. 2010). Classification of these images is particularly challenging because they are not preprocessed or normalized. An R script and instructions for constructing the dataset we analyzed are provided in the supplementary material, and the original images are available online (*https://labicvl.github.io/ges_db.htm*).

Using leave-one-out cross-validation, our method correctly classified 72 of 80 images when selecting tuning parameters for use in (3) by minimizing the five-fold cross-validation misclassification rate. As displayed in Table 3, our method performed better than the model-based competitors we considered. Our method had the same classification accuracy as neural networks with decay 5×10^{-4} and size four fit using the R package `nnet` (Venables and Ripley 2002).

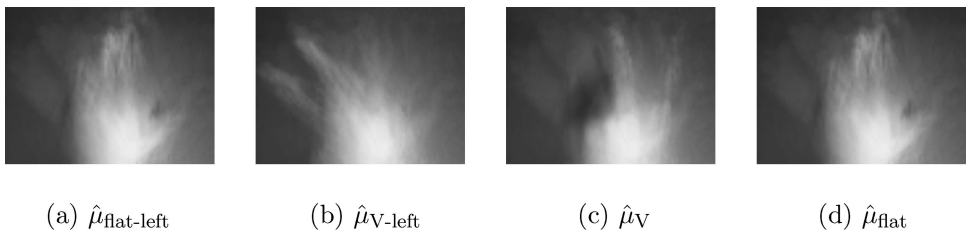
Because we used a model-based approach, we estimated mean images for each of the four positions jointly. We refit (3) using the complete dataset with the tuning parameter pair that minimized leave-one-out classification accuracy, which correctly classified 74 of 80 images. In Figure 5(a)–5(g), we display the estimated mean images from our fitted model and three absolute pairwise differences. The absolute difference between the flat-left and V-left positions in Figure 5(e) shows pixels estimated to have nonzero mean differences are located between the spread fingers on the left-hand side of the image. The estimated nonzero mean differences between the V-centered and V-left positions displayed in Figure 5(f) mainly capture the shadows that appear behind the V-centered position images, which do not appear in the V-left position images.

6. Extensions

6.1. Quadratic Discriminant Analysis

Our method naturally extends to the quadratic discriminant analysis model, where one assumes

$$\text{vec}(X) \mid Y = j \sim \text{N}_{rc}\left\{\text{vec}\left(\mu_{*j}\right), \Sigma_{*j}\right\}, \quad j = 1, \ldots, J,$$



(a) $\hat{\mu}_{\text{flat-left}}$ (b) $\hat{\mu}_{\text{V-left}}$ (c) $\hat{\mu}_{\text{V}}$ (d) $\hat{\mu}_{\text{flat}}$

(e) $|\hat{\mu}_{\text{flat-left}} - \hat{\mu}_{\text{V-left}}|$ (f) $|\hat{\mu}_{\text{V}} - \hat{\mu}_{\text{V-left}}|$ (g) $|\hat{\mu}_{\text{flat}} - \hat{\mu}_{\text{V-left}}|$

Figure 5. (a)–(d) are the mean images estimated using (3) for each of the four hand positions: flat-left, V-left, V-centered, and flat-centered, respectively. (e)–(g) show the estimated absolute pairwise differences between various mean matrices of the four positions. White pixels correspond to zero and darker pixels indicate a larger magnitude in (e)–(g).

where $\Sigma_{*j} \in \mathbb{S}_+^{rc}$ is the covariance matrix for the jth response category. To generalize (2), one can assume either (i) $\Sigma_{*j}^{-1} = \Delta_{*j} \otimes \Phi_{*j}$, (ii) $\Sigma_{*j}^{-1} = \Delta_{*j} \otimes \Phi_*$, or (iii) $\Sigma_{*j}^{-1} = \Delta_* \otimes \Phi_{*j}$. Our algorithms could be modified to accommodate these cases.

6.2. Classification with Tensor-Valued Predictors

The assumption (2) and estimator (3) can also be generalized to cases where the predictor is a multi-dimensional array of order three or more (i.e., a "tensor"), such as in fMRI or video data. In this case, where $X \in \mathbb{R}^{d_1 \times \cdots \times d_q}$, we can generalize the assumption (2) to the matrix $\Sigma_* \in \mathbb{S}_+^K$, where $K = \prod_{i=1}^q d_i$ so that

$$\Sigma_*^{-1} = \Xi_q \otimes \cdots \otimes \Xi_1, \tag{14}$$

with $\Xi_l \in \mathbb{S}_+^{d_l}$ for $l = 1, \ldots, q$. Using (14), (1) becomes the array-normal distribution (Hoff 2011; Manceur and Dutilleul 2013). Algorithm 1 can be extended using the ideas developed in Section 3.2 and our alternating minimization algorithm could be generalized by replacing the matrix product in the right-hand side of (12) with a tensor multiplication.

Supplementary Materials

Appendix: Includes simulations comparing our method to the methods proposed by Zhong and Suslick (2015) and vector-valued sparse linear discriminant methods; simulations illustrating the efficiency gained by joint estimation of the μ_{*j}'s, Δ_*, and Φ_* using (3); and simulations investigating the sensitivity of (3) to the choice of weights.

Code: Includes R scripts to create the real datasets we analyze in Section 5, and to reproduce the simulation results.

MatrixLDA: An R package implementing our method, along with auxiliary functions for prediction and tuning parameter selection.

Acknowledgments

The authors thank the associate editor and referees for helpful comments.

Funding

This research was supported in part by the Doctoral Dissertation Fellowship from the University of Minnesota and the National Science Foundation grant DMS-1452068.

References

Allen, G. I., and Tibshirani, R. J. (2010), "Transposable Regularized Covariance Models with an Application to Missing Data Imputation," *The Annals of Applied Statistics*, 4, 764–790. [12]

Baranzini, S. E., Mousavi, P., Rio, J., Caillier, S. J., Stillman, A., Villoslada, P., Wyatt, M. M., Comabella, M., Greller, L. D., Somogyi, R., Montalban, X., and Oksenberg, J. R. (2004), "Transcription-Based Prediction of Response to IFNβ using Supervised Computational Methods," *PLoS Biology*, 3, e2. [11,19]

Bickel, P. J., and Levina, E. (2004), "Some Theory for Fisher's Linear Discriminant Function,'Naive Bayes', and Some Alternatives When There are Many More Variables than Observations," *Bernoulli*, 10, 989–1010. [11]

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011), "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning*, 3, 1–122. [13]

Buuren, S. van, and Groothuis-Oudshoorn, K. (2010), "Mice: Multivariate Imputation by Chained Equations in R," *Journal of Statistical Software*, 45, 1–68. [19]

Chi, E. C., and Lange, K. (2015), "Splitting Methods for Convex Clustering," *Journal of Computational and Graphical Statistics*, 24, 994–1013. [13,14]

Clemmensen, L., Hastie, T., Witten, D., and Ersbøll, B. (2011), "Sparse Discriminant Analysis," *Technometrics*, 53, 406–413. [18]

Dutilleul, P. (1999), "The MLE Algorithm for the Matrix Normal Distribution," *Journal of Statistical Computation and Simulation*, 64, 105–123. [12,14,15]

Friedman, J. H., Hastie, T. J., and Tibshirani, R. J. (2008), "Sparse Inverse Covariance Estimation with the Graphical Lasso," *Biostatistics*, 9, 432–441. [13,15]

Goldstein, T., O'Donoghue, B., Setzer, S., and Baraniuk, R. (2014), "Fast Alternating Direction Optimization Methods," *SIAM Journal on Imaging Sciences*, 7, 1588–1623. [13,14]

Guo, J. (2010), "Simultaneous Variable Selection and Class Fusion for High-Dimensional Linear Discriminant Analysis," *Biostatistics*, 11, 599–608. [12,13,16,17]

Gupta, A. K., and Nagar, D. K. (2000), *Matrix Variate Distributions*, Boca Raton, FL: Chapman and Hall/CRC Press. [11]

Helwig, N. E. (2015), "eegkit: Toolkit for Electroencephalography Data," R Package Version 1.0-2. [19]

Hoff, P. D. (2011), "Separable Covariance Arrays via the Tucker Product, with Applications to Multivariate Relational Data," *Bayesian Analysis*, 6, 179–196. [12,21]

Hung, H., and Wang, C.-C. (2013), "Matrix Variate Logistic Regression Model with Application to EEG Data," *Biostatistics*, 14, 189–202. [11]

Hunter, D. R., and Li, R. (2005), "Variable Selection using MM Algorithms," *The Annals of Statistics*, 33, 1617–1642. [13]

Ingber, L. (1998), "Statistical Mechanics of Neocortical Interactions: Training and Testing Canonical Momenta Indicators of EEG," *Mathematical and Computer Modelling*, 27, 33–64. [11]

Kim, T.-K., and Cipolla, R. (2009), "Canonical Correlation Analysis of Video Volume Tensors for Action Categorization and Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31, 1415–1428. [20]

Lee, W., and Liu, Y. (2012), "Simultaneous Multiple Response Regression and Inverse Covariance Matrix Estimation via Penalized Gaussian Maximum Likelihood," *Journal of Multivariate Analysis*, 111, 241–255. [12]

Leng, C., and Tang, C. Y. (2012), "Sparse Matrix Graphical Models," *Journal of the American Statistical Association*, 107, 1187–1200. [12]

Li, B., Kim, M. K., and Altman, N. (2010), "On Dimension Folding of Matrix-or Array-Valued Statistical Objects," *The Annals of Statistics*, 38, 1094–1121. [11,18,19]

Li, L., and Zhang, X. (2017), "Parsimonious Tensor Response Regression," *Journal of the American Statistical Association*, 112, 1131–1146. [12]

Li, M., and Yuan, B. (2005), "2D-LDA: A Statistical Linear Discriminant Analysis for Image Matrix," *Pattern Recognition Letters*, 26, 527–532. [11]

Lyu, T., Lock, E. F., and Eberly, L. E. (2017), "Discriminating Sample Groups with Multi-Way Data," *Biostatistics*, 18, 434–450. [20]

Mai, Q., Yang, Y., and Zou, H. (2018), "Multiclass Sparse Discriminant Analysis," *Statistica Sinica*. doi:10.5705/ss.202016.0117. [18]

Manceur, A. M., and Dutilleul, P. (2013), "Maximum Likelihood Estimation for the Tensor Normal Distribution: Algorithm, Minimum Sample Size, and Empirical Bias and Dispersion," *Journal of Computational and Applied Mathematics*, 239, 37–49. [21]

Mitchell, M. W., Genton, M. G., and Gumpertz, M. L. (2006), "A Likelihood Ratio Test for Separability of Covariances," *Journal of Multivariate Analysis*, 97, 1025–1043. [15]

Pan, R., Wang, H., and Li, R. (2016), "Ultrahigh-Dimensional Multiclass Linear Discriminant Analysis by Pairwise Sure Independence Screening," *Journal of the American Statistical Association*, 111, 169–179. [16,17]

Parikh, N., and Boyd, S. (2014), "Proximal Algorithms," *Foundations and Trends in Optimization*, 1, 127–239. [14]

Pau, G., Fuchs, F., Sklyar, O., Boutros, M., and Huber, W. (2010), "EBImage—An R package for Image Processing with Applications to Cellular Phenotypes," *Bioinformatics*, 26, 979–981. [20]

R Core Team (2017), *R: A Language and Environment for Statistical Computing*, Vienna, Austria: R Foundation for Statistical Computing. [13]

Roś, B., Bijma, F., de Munck, J. C., and de Gunst, M. C. (2016), "Existence and Uniqueness of the Maximum Likelihood Estimator for Models with a Kronecker Product Covariance Structure," *Journal of Multivariate Analysis*, 143, 345–361. [11]

Rothman, A. J., Levina, E., and Zhu, J. (2010), "Sparse Multivariate Regression with Covariance Estimation," *Journal of Computational and Graphical Statistics*, 19, 947–962. [12]

Tseng, P. (1991), "Applications of a Splitting Algorithm to Decomposition in Convex Programming and Variational Inequalities," *SIAM Journal on Control and Optimization*, 29, 119–138. [12,13,14]

Tsiligkaridis, T., Hero, A. O., and Zhou, S. (2012), "Kronecker Graphical Lasso," in *2012 IEEE Statistical Signal Processing Workshop (SSP)*, IEEE, pp. 884–887. [12]

Venables, W. N., and Ripley, B. D. (2002), *Modern Applied Statistics with S* (4th ed.), New York: Springer. [20]

Witten, D. M., Friedman, J. H., and Simon, N. (2011), "New Insights and Faster Computations for the Graphical Lasso," *Journal of Computational and Graphical Statistics*, 20, 892–900. [13]

Witten, D. M., and Tibshirani, R. (2011), "Penalized Classification using Fisher's Linear Discriminant," *Journal of the Royal Statistical Society*, Series B, 73, 753–772. [18]

Xia, Y., and Li, L. (2017), "Hypothesis Testing of Matrix Graph Model with Application to Brain Connectivity Analysis," *Biometrics*, 73, 780–791. [12]

Xu, P., Zhu, J., Zhu, L., and Li, Y. (2015), "Covariance-Enhanced Discriminant Analysis," *Biometrika*, 102, 33–45. [11,12,13,15,19]

Yin, J., and Li, H. (2011), "A Sparse Conditional Gaussian Graphical Model for Analysis of Genetical Genomics Data," *The Annals of Applied Statistics*, 5, 2630–2650. [12]

Zhang, Y., and Schneider, J. G. (2010), "Learning Multiple Tasks with a Sparse Matrix-Normal Penalty," in *Advances in Neural Information Processing Systems*, pp. 2550–2558. [12]

Zhong, W., and Suslick, K. S. (2015), "Matrix Discriminant Analysis With Application to Colorimetric Sensor Array Data," *Technometrics*, 57, 524–534. [11,18]

Zhou, H., and Li, L. (2014), "Regularized Matrix Regression," *Journal of the Royal Statistical Society*, Series B, 76, 463–483. [11,18,19]

Zhou, S. (2014), "Gemini: Graph Estimation with Matrix Variate Normal Instances," *The Annals of Statistics*, 42, 532–562. [12,18]

Zhu, Y. (2017), "An augmented ADMM Algorithm with Application to the Generalized Lasso Problem," *Journal of Computational and Graphical Statistics*, 26, 195–204. [14]