Covariance-adjusted, Sparse, Reduced-Rank Regression with Adjustment for Confounders

by

Xuefei Yang

B.Sc., Simon Fraser University, 2018

Project Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
Department of Statistics and Actuarial Science
Faculty of Science

Declaration of Committee

Name:	**Xuefei Yang**
Degree:	**Master of Science**
Thesis title:	**Covariance-adjusted, Sparse, Reduced-Rank Regression with Adjustment for Confounders**
Committee:	**Chair:** X. Joan Hu Professor, Statistics and Actuarial Science
	Brad McNeney Supervisor Associate Professor, Statistics and Actuarial Science
	Lloyd Elliott Committee Member Assistant Professor, Statistics and Actuarial Science
	Jinko Graham Examiner Professor, Statistics and Actuarial Science

Abstract

There is evidence that common genetic variation in the gene *NEDD9* is associated with developing Alzheimer's Disease (AD). In this project, we study the relationship between brain-imaging biomarkers of AD and the gene *NEDD9* while adjusting for the effects of genetic population structure. The data used in this project, collected by the Alzheimer's Disease Neuroimaging Initiative (ADNI), consists of magnetic resonance imaging (MRI) measures of 56 brain regions of interest for 200 cognitively normal people and genetic data on Single Nucleotide Polymorphisms (SNPs) obtained from 33 candidate genes for AD. The standard solution to such a multiple response problem is separate simple linear regression models. Such an approach neglects correlations between 56 brain areas and possible sparsity in the SNP effects. In this project we review a sparse and covariance adjusted reduced-rank regression approach that can select significant predictors and estimate covariance simultaneously, and extend the approach to adjust for confounding variables. We apply the proposed algorithm to the ADNI data, and also simulated data.

Keywords: Alzheimer's Disease; Brain-imaging data; gene NEDD9; Principle Component Analysis; Multiple-response problem; Confounder adjustment;

Acknowledgements

I would like to express my heartfelt gratitude to my primary supervisor, Dr. Brad McNeney, his patient guidance and enthusiastic encouragement guided me throughout this project. My completion of this project could not have been accomplished without the support of him. My grateful thanks are also extended to Mr.Pulindu Ratnasekera, who helped me generate the genotypes data and gave me suggestions on my code.

Many thanks to all of my classmates for their kind support during my MSc study.

Lastly, thanks to my parents, for always being there for me. Thanks to my sister and brother, for encouraging and supporting me whenever I needed them.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Alzheimer's Disease (AD) is the most common cause of dementia and there is a growing number of elderly people living with AD. Currently, treatment options for AD are limited. Genetic variants are thought to affect a person's risk of developing AD. For example, Li et al. (2008) found that common genetic variants in gene *NEDD9* (Neural precursor cell expressed developmentally down-regulated protein 9) are associated with developing AD. In this project we explore the relationship between brain-imaging biomarkers of AD and genetic variants in *NEDD9*. The data used in this project was collected by the researchers from the Alzheimer's Disease Neuroimaging Initiative (ADNI), which involves two different studies, called ADNI-1 and ADNI-GO/2. The data contain magnetic resonance imaging (MRI) measurements of 56 brain regions of interest for each of these subjects and their corresponding SNP genotypes obtained from 33 candidate genes for AD. Our interest is in genetic variation that predicts structural differences in the brain *before* subjects experience memory loss. Hence we use data on cognitively normal subjects from ADNI. The brain-imaging phenotypes will be the response variables in our model and the SNP genotypes from gene *NEDD9* will be the explanatory variables.

Genetic variation in the human genome differs among individuals in different populations due to genetic population structure. When such structure is correlated with a phenotype, it can confound genetic associations with the phenotype. Thus, population structure is an important confounding variable in genetic association studies. Menozzi et al. (1978) established the use of Principle Component Analysis (PCA) to summarize individual genetic variation across different regions/populations. PCA of a set of variables looks for linear combinations of these variables (PCs) that capture maximal variance. Thus, the PCs can help us characterize different populations. Adding a number of important PCs to our model can mitigate the effect of population structure.

Our goal of investigating associations between multiple response variables (brain phenotypes) and multiple explanatory variables is an example of a multiple-response problem. The naive approach, separate linear regression models, ignores possible interrelations between the response variables and the correlation among the errors [Chen and Huang (2012)]. In this project, we hope to find SNPs that can help explain the MRI measurements and also exploit the correlation between the nearby brain regions of interest within a subject. We would also like to consider the effect of confounding variables. Failing to control for confounders can distort the true relationship and lead to misleading results.

In this project, we apply a multiple-response method called Sparse Reduced-Rank Regression with Covariance Estimation (Cov-SRRR) to improve predictive power and interpretability. The Cov-SRRR model improves the naive model in the sense that it performs shrinkage and variable selection of predictors and covariances simultaneously, and exploits the correlation between the response variables.

The standard method for including confounding variables is to first adjust both the response and predictor variables for the confounder, and then perform Cov-SRRR on the adjusted variables. Here the adjusted response and predictor variables are residuals from regressions on the confounding variables. The rationale for this approach is ordinary least squares, where such a residuals-on-residuals regression yields the least squares estimates of the coefficients of the predictor variables (Weisberg (2013)). However, in light of the shrinkage and variable selection of Cov-SRRR the correspondence between an analysis of confounder-adjusted response and predictor variables and *direct* modeling and adjustment for confounding variables is less clear. In this project we extend Cov-SRRR to include direct modeling and adjustment for potential confounding variables (Cov-Con-SRRR). We apply Cov-Con-SRRR to both simulated and real data, and compare our results to those obtained from a standard confounder-adjusted Cov-SRRR.

Chapter 2

Methodology

In this chapter, we develop the model and algorithm for Cov-SRRR with adjustment for confounders (Cov-Con-SRRR). For a sample of size n, let $\boldsymbol{Y} = \{Y_1, ..., Y_q\}$ be an $n \times q$ matrix of q response variables on the n subjects, let $\boldsymbol{X} = \{X_1, ..., X_p\}$ be an $n \times p$ matrix of p explanatory variables, and let $\boldsymbol{Z} = \{Z_1, ..., Z_h\}$ be an $n \times h$ matrix of h confounding variables.

2.1 Naive Approach

Consider the simple multiple-response linear regression:

$$\mathbf{Y}_j = \sum_{i=1}^{p} \mathbf{X}_i \mathbf{C}_{ij} + \sum_{i=1}^{h} \mathbf{Z}_i \mathbf{D}_{ij} + \epsilon_j, \tag{2.1.1}$$

where j=1,2,. . . , q, and Y_j is the j^{th} response vector for all n individuals. We center the response variables, predictors, and the confounding variables to have mean zero so that we can omit the intercept term. The error terms are denoted $(\epsilon_1, ..., \epsilon_q)^T$; we discuss their distribution below.

The model in the matrix form is:

$$\mathbf{Y} = \mathbf{XC} + \mathbf{ZD} + \mathbf{E} \tag{2.1.2}$$

where $\mathbf{Y}$ is the $n \times q$ response matrix, $\mathbf{X}$ is the $n \times p$ predictor matrix, $\mathbf{Z}$ is the $n \times h$ confounder matrix, $\mathbf{C}$ and $\mathbf{D}$ are the $p \times q$ and $h \times q$ matrices of coefficients, respectively, and $\mathbf{E}$ is the $n \times q$ error matrix. The rows of $\mathbf{E}$ are assumed to be independent and identically distributed as a multivariate normal distribution with mean zero and variance Σ_e.

We can combine the predictor matrix X and confounder matrix Z into one $n \times (p+h)$ matrix F, so that the model becomes:

$$\mathbf{Y} = \mathbf{FG} + \mathbf{E}$$

where $\mathbf{G}$ is a $(p+h) \times q$ coefficient matrix. Specifically,

$$\mathbf{Y} = \begin{bmatrix} X_{11} & \dots & X_{1p} & Z_{11} & \dots & Z_{1h} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_{n1} & \dots & X_{np} & Z_{n1} & \dots & Z_{nh} \end{bmatrix} \begin{bmatrix} C_{11} & \dots & C_{1q} \\ \vdots & \vdots & \vdots \\ C_{p1} & \dots & C_{pq} \\ D_{11} & \dots & D_{1q} \\ \vdots & \vdots & \vdots \\ D_{h1} & \dots & D_{hq} \end{bmatrix} + \mathbf{E}$$

An estimate of the coefficient matrix $\mathbf{G}$ is obtained by solving the following Ordinary Least Squares (OLS) problem:

$$\min_G \|\mathbf{Y} - \mathbf{FG}\|^2 = \min_G \sum_{j=1}^{q} \sum_{i=1}^{n} \left(Y_{ij} - \sum_{k=1}^{p} F_{ik} G_{kj} \right)^2$$

where $\|\cdot\|$ denotes the Frobenius norm. Recall that for a matrix A, its Frobenius norm $\|A\| = \sqrt{\sum_i \sum_j a_{ij}^2} = \sqrt{tr(A^T A)}$. The OLS solution has the form:

$$\hat{\mathbf{G}}_{OLS} = \begin{bmatrix} \hat{\mathbf{C}}_{OLS} \\ \hline \hat{\mathbf{D}}_{OLS} \end{bmatrix} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{Y} \tag{2.1.3}$$

We say the OLS estimates of the coefficients $\mathbf{C}$ and $\mathbf{D}$ are naive because they ignore the fact that the response variables $(\mathbf{Y}_1, \dots, \mathbf{Y}_j)$ might correlate with each other. What's more, there does not exist unique solution when p is greater than n.

In genetic studies, we may collect data on thousands of genetic markers. Thus, to handle the dimensionality of the predictor matrix and take into account the possible interrelationships between $\mathbf{Y}_j$'s, we will introduce an improved method known as Reduced-Rank Regression (RRR) in the following section.

2.2 Reduced-Rank Regression (RRR)

Recall the model

$$\mathbf{Y} = \mathbf{XC} + \mathbf{ZD} + \mathbf{E}$$

with unknown parameters $\mathbf{C}$, $\mathbf{D}$ and Σ_e, where Σ_e is the variance of the rows of $\mathbf{E}$. In order to conform with the general RRR model in matrix format, we rewrite the above as:

$$\mathbf{Y} - \mathbf{ZD} = \mathbf{XC} + \mathbf{E}.$$

The negative log-likelihood function, up to a constant [Velu and Reinsel (2013)], is:

$$\begin{aligned} l(\mathbf{C},\, \mathbf{D}, \mathbf{\Sigma}_e) &= -log|\mathbf{\Sigma}_e^{-1}| + \frac{1}{n}\sum_{i=1}^{n}[(\mathbf{Y}_i - \mathbf{Z}_i\mathbf{D}_i - \mathbf{X}_i\mathbf{C}_i)^T\mathbf{\Sigma}_e^{-1}(\mathbf{Y}_i - \mathbf{Z}_i\mathbf{D}_i - \mathbf{X}_i\mathbf{C}_i)] \\ &= -log|\mathbf{\Sigma}_e^{-1}| + \frac{1}{n}tr[(\mathbf{Y} \text{ - } \mathbf{ZD} \text{ - } \mathbf{XC})\mathbf{\Sigma}_e^{-1}(\mathbf{Y} \text{ - } \mathbf{ZD} \text{ - } \mathbf{XC})^T], \end{aligned}$$

where $\left|\mathbf{\Sigma}_e^{-1}\right|$ is the determinant of $\mathbf{\Sigma}_e^{-1}$. Let $\mathbf{\Omega} = \mathbf{\Sigma}_e^{-1}$ be the precision matrix (inverse-covariance matrix) . After substituting $\mathbf{\Omega}$ and using the cyclic property of the trace, the negative log-likelihood function of ($\mathbf{C},\, \mathbf{D},\, \mathbf{\Omega}$) in the RRR model is

$$\frac{1}{n}tr[(\mathbf{Y} \text{ - } \mathbf{ZD} - \mathbf{XC})^T(\mathbf{Y} \text{ - } \mathbf{ZD} - \mathbf{XC})\mathbf{\Omega}] - log\,|\mathbf{\Omega}|\,. \tag{2.2.1}$$

The RRR model imposes a rank constraint on the matrix $\mathbf{C}$ in such a way that the improved model has reduced the dimension with minimal loss of information [Izenman (1975)].

Suppose

$$rank(\mathbf{C}) = r, \text{ where } r \leq min(p, q),$$

then there exists a $q \times r$ matrix $\mathbf{A}$ and a $p \times r$ matrix $\mathbf{B}$ such that:

$$\mathbf{C} = \mathbf{B}\mathbf{A}^T$$

Under this rank constraint the model becomes:

$$\mathbf{Y\text{-}ZD} = (\mathbf{XB})\mathbf{A}^T + \mathbf{E}. \tag{2.2.2}$$

The negative log-likelihood function of $(\mathbf{B},\ \mathbf{A},\ \mathbf{D},\ \boldsymbol{\Omega})$ in (2.2.2) is:

$$\frac{1}{n}tr[(\mathbf{Y}\ \text{-}\ \mathbf{ZD} - \mathbf{XBA}^T)^T(\mathbf{Y}\ \text{-}\ \mathbf{ZD} - \mathbf{XBA}^T)\boldsymbol{\Omega}] - log\,|\boldsymbol{\Omega}|\,. \tag{2.2.3}$$

$\mathbf{XB}$ can be viewed as a new predictor matrix with only r components, and $\mathbf{A}^T$ as the coefficient matrix. In other words, $\mathbf{XB}$ represents r linear combinations of the predictor variables that help explain the variation in $\mathbf{Y\text{-}ZD}$. Under this reduced-rank setting, we decrease the number of free parameters in $\mathbf{C}$ from pq to $(p+q)r$.

The coefficient matrices $\mathbf{A}$ and $\mathbf{B}$ are not unique. For any $r \times r$ invertible matrix $\mathbf{P}$, $\mathbf{BA}^T = \mathbf{BPP}^{-1}\mathbf{A}^T = \mathbf{WQ}^T = \mathbf{C}$, where $\mathbf{W} = \mathbf{BP}$ and $\mathbf{Q}^T = \mathbf{P}^{-1}\mathbf{A}^T$ are $p \times r$ and $r \times q$ matrices, respectively. Thus, $\mathbf{WQ}^T = \mathbf{C}$ is another decomposition of $\mathbf{C}$ [Ruan (2019)]. Chen and Huang (2016) suggest to add the constraints that $\mathbf{A}^T\boldsymbol{\Omega}\mathbf{A} = \mathbf{I}_r$ and $\mathbf{B}^T\mathbf{S}_x\mathbf{B}$ is diagonal to ensure identifiability, where $\mathbf{S}_x$ is the sample covariance matrix of the predictor $\mathbf{X}$.

Assume r, $\mathbf{D}$ and $\boldsymbol{\Omega}$ are known. Then the RRR optimization problem is:

$$\begin{aligned}&\min_{\mathbf{A},\mathbf{B}} \frac{1}{n}tr[(\mathbf{Y\text{-}ZD} - \mathbf{XBA}^T)^T(\mathbf{Y\text{-}ZD} - \mathbf{XBA}^T)\boldsymbol{\Omega}] \\ &s.t.\ \mathbf{A}^T\boldsymbol{\Omega}\mathbf{A} = \mathbf{I}_r \ \text{ and } \ \mathbf{B}^T\mathbf{S}_x\mathbf{B} \text{ is diagonal}\end{aligned} \tag{2.2.4}$$

We do not impose a rank constraint on the confounder coefficient matrix $\mathbf{D}$; we include confounders in our model to ensure correct estimates of $\mathbf{A}$ and $\mathbf{B}$, and do not want to impose any shrinkage on $\mathbf{D}$ that could compromise estimation of $\mathbf{A}$ and $\mathbf{B}$. Estimation of $\mathbf{D}$ will be discussed in section (2.4).

The RRR model helps achieve the goal of dimension reduction by introducing the latent factors (**XB**) that explain the main variation in **Y - ZD** [Ruan (2019)], but it does not support variable selection and covariance estimation. In the next section we discuss an approach called Cov-SRRR that does.

2.3 SRRR with Covariance Estimation

Compared to the RRR model, the Cov-SRRR model brings in two regularization penalties on the negative log-likelihood function. A row-wise penalty on the matrix **B** ensures variable selection, while an element-wise LASSO penalty on the inverse-covariance matrix **Ω** stabilizes covariance estimation.

To introduce sparsity, we replace the constraint on **B** with a row-wise penalty:

$$\sum_{j=1}^{p} \lambda_j \sqrt{\sum_{i=1}^{r} (\mathbf{B}_i^j)^2} = \sum_{j=1}^{p} \lambda_j \left\| \mathbf{B}^j \right\|_2 \tag{2.3.1}$$

where $\mathbf{B}^j$ is the j^{th} row of the matrix $\mathbf{B}$, λ_j is a tuning parameter for the j^{th} row of $\mathbf{B}$, and $\|\cdot\|_2$ is the L^2 norm of a row vector. For computational efficiency, we set all $\lambda_j' s$ to be equal. Thus, for fixed $\mathbf{D}$ and Ω, we consider the following objective function:

$$\min_{\mathbf{A},\mathbf{B}} \frac{1}{n} tr[(\mathbf{Y}\text{-}\mathbf{ZD} - \mathbf{XBA}^T)^T (\mathbf{Y}\text{-}\mathbf{ZD} - \mathbf{XBA}^T)\boldsymbol{\Omega}] + \lambda_1 \sum_{j=1}^{p} \left\| \mathbf{B}^j \right\|_2 \tag{2.3.2}$$

$$\text{s.t. } \mathbf{A}^T \boldsymbol{\Omega} \mathbf{A} = \mathbf{I}_r$$

By penalizing rows of the coefficient matrix $\mathbf{B}$, we cause shrinkage of the coefficients $\mathbf{B}^j$ towards zero. An all-zero vector is equivalent to excluding the corresponding predictor variable $\mathbf{X}_j$ [Chen and Huang (2012)].

The solution of $\mathbf{A}$ and $\mathbf{B}$ is unique up to an $r \times r$ orthogonal matrix [Chen and Huang (2016)]. Specifically, for any orthogonal matrix $\mathbf{W}$, $\mathbf{WW}^T = \mathbf{I}_r$ so that, $\mathbf{BA}^T = \mathbf{BWW}^T\mathbf{A}^T = \mathbf{B}^*\mathbf{A}^{*T}$, where $\mathbf{B}^* = \mathbf{BW}$ and $\mathbf{A}^* = \mathbf{AW}$.

For fixed $\mathbf{A}$ and $\mathbf{B}$, the inverse covariance matrix $\boldsymbol{\Omega}$ is estimated by penalized maximum likelihood. We apply the following LASSO penalty on $\boldsymbol{\Omega}$ [Chen and Huang

(2016)]:

$$\lambda_2 \sum_{j \neq j'} \left|\omega_{jj'}\right| \tag{2.3.3}$$

where $\omega_{jj'}$ is the (j, j') entry in $\mathbf{\Omega}$, $j' \neq j$, λ_2 is a tuning parameter and $|\cdot|$ denotes absolute value. This penalty encourages sparsity by penalizing all the off-diagonal elements of $\mathbf{\Omega}$. Thus, for fixed $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{D}$ we consider the following objective function:

$$\min_{\Omega} \quad tr(\mathbf{S}_e\mathbf{\Omega}) - log\left|\mathbf{\Omega}\right| + \lambda_2 \sum_{j \neq j'} \left|\omega_{jj'}\right| \tag{2.3.4}$$

where $\mathbf{S}_e = \frac{1}{n}(\mathbf{Y\text{-}ZD} - \mathbf{XBA}^T)^T(\mathbf{Y\text{-}ZD} - \mathbf{XBA}^T)$ is the sample covariance matrix of $\mathbf{E}$.

Combining the above two sparsity-inducing penalties (2.3.1) and (2.3.3) to the RRR model, and considering $\mathbf{D}$ to be a free parameter, the penalized negative log-likelihood function becomes:

$$\begin{aligned} \min_{A,B,D,\Omega} \quad & \frac{1}{n}tr[(\mathbf{Y}\text{ - }\mathbf{ZD} - \mathbf{XBA}^T)^T(\mathbf{Y}\text{ - }\mathbf{ZD} - \mathbf{XBA}^T)\mathbf{\Omega}] \\ & - log\left|\mathbf{\Omega}\right| + \lambda_1 \sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2 + \lambda_2 \sum_{j \neq j'} \left|\omega_{jj'}\right|, \\ & s.t. \mathbf{A}^T\mathbf{\Omega}\mathbf{A} = \mathbf{I}_r \end{aligned} \tag{2.3.5}$$

2.4 Computational Algorithm

In order to solve the complex optimization problem in (2.3.5), we divide the objective function into two parts, with just one penalty term in each part. To be more specific, the Cov-Con-SRRR algorithm algorithm for minimizing the penalized negative log-likelihood iterates between updating $\mathbf{\Omega}$ for fixed ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$) and updating ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$) for fixed $\mathbf{\Omega}$ [Chen and Huang (2016)].

For fixed ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$), the objective function for solving $\mathbf{\Omega}$ is:

$$\begin{aligned} \min_{\Omega} \quad & \frac{1}{n}tr[(\mathbf{Y\text{-}ZD} - \mathbf{XBA}^T)^T(\mathbf{Y\text{-}ZD} - \mathbf{XBA}^T)\mathbf{\Omega}] \\ & - log\left|\mathbf{\Omega}\right| + \lambda_2 \sum_{j \neq j'} \left|\omega_{jj'}\right| \end{aligned} \tag{2.4.1}$$

According to Chen and Huang (2016), the above problem can be solved by using the dual-primal graphical LASSO (DP-GLASSO) algorithm [Mazumder and Hastie (2012)]. However, the objective function in DP-GLASSO includes a penalty on *all* elements of the precision matrix $\mathbf{\Omega}$, where as the penalty term in (2.4.1) indicates that we are not supposed to penalize the diagonal elements of $\mathbf{\Omega}$. Furthermore, Halani (2016) found that there exists convergence problems when we use the DP-GLASSO algorithm. In this project, instead of the DP-GLASSO, we use a flexible R package called **CVXR** that is designed for convex optimization problems, including GLASSO, to solve the problem in (2.4.1).

Given $\mathbf{\Omega}$, we update $\mathbf{A}$**,** $\mathbf{B}$ and $\mathbf{D}$ by solving the following objective function:

$$\begin{aligned}\min_{A,B,D}\ & \frac{1}{n}tr[(\mathbf{Y}\text{ - }\mathbf{ZD}-\mathbf{XBA}^T)^T\mathbf{\Omega}(\mathbf{Y}\text{ - }\mathbf{ZD}-\mathbf{XBA}^T)] \\ & -log\,|\mathbf{\Omega}|+\lambda_1\sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2 \\ & s.t.\mathbf{A}^T\mathbf{\Omega}\mathbf{A}=\mathbf{I}_r\end{aligned} \tag{2.4.2}$$

Let $\tilde{\mathbf{A}}=\mathbf{\Omega}^{\frac{1}{2}}\mathbf{A}$, then the objective function becomes:

$$\begin{aligned}&\frac{1}{n}tr[((\mathbf{Y}\text{-}\mathbf{ZD})\mathbf{\Omega}^{\frac{1}{2}}-\mathbf{XB}\tilde{\mathbf{A}}^T)^T((\mathbf{Y}\text{-}\mathbf{ZD})\mathbf{\Omega}^{\frac{1}{2}}-\mathbf{XB}\tilde{\mathbf{A}}^T)]+\lambda_1\sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2 \\ &=\frac{1}{n}\left\|(\mathbf{Y}\text{-}\mathbf{ZD})\mathbf{\Omega}^{\frac{1}{2}}-\mathbf{XB}\tilde{\mathbf{A}}^T)\right\|^2+\lambda_1\sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2 \\ &=\frac{1}{n}\left\|(\tilde{\mathbf{Y}}-\mathbf{Z}\tilde{\mathbf{D}})-\mathbf{XB}\tilde{\mathbf{A}}^T)\right\|^2+\lambda_1\sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2,\end{aligned}$$

where $\tilde{\mathbf{D}}=\mathbf{D}\mathbf{\Omega}^{\frac{1}{2}}$ and $\tilde{\mathbf{Y}}=\mathbf{Y}\mathbf{\Omega}^{\frac{1}{2}}$, and the constraint becomes $\tilde{\mathbf{A}}^T\tilde{\mathbf{A}}=\mathbf{I}_r$. Thus, (2.4.2) becomes:

$$\begin{aligned}\min_{\tilde{A},B,\tilde{D}}\ & \frac{1}{n}\left\|(\tilde{\mathbf{Y}}-\mathbf{Z}\tilde{\mathbf{D}})-\mathbf{XB}\tilde{\mathbf{A}}^T)\right\|^2+\lambda_1\sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2 \\ & s.t.\tilde{\mathbf{A}}^T\tilde{\mathbf{A}}=\mathbf{I}_r\end{aligned} \tag{2.4.3}$$

For fixed $\mathbf{D}$, the above optimization problem is a standard SRRR of the residuals $\mathbf{Y}$ **-** $\mathbf{ZD}$ on $\mathbf{X}$ and the solution can be obtained by iterating between solutions of $\tilde{\mathbf{A}}$ or

$\mathbf{B}$ holding the other fixed. Extending this idea, the solution to (2.4.3) can be obtained using a block-wise coordinate descent (BCD) algorithm[Nutini et al. (2017)]; that is we update $(\tilde{\mathbf{A}}, \mathbf{B}, \tilde{\mathbf{D}})$ one at a time while leaving the others unchanged.

First, update $\tilde{\mathbf{A}}$ for fixed $(\mathbf{B}, \tilde{\mathbf{D}})$. Equation (2.4.3) can be viewed as

$$\begin{aligned} &\min_{\tilde{A}} \left\| (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T \right\|^2, \\ &s.t. \tilde{\mathbf{A}}^T \tilde{\mathbf{A}} = \mathbf{I}_r \end{aligned} \tag{2.4.4}$$

and the above (2.4.4) optimization problem is known as a orthogonal Procrustes problem with solution $\tilde{\mathbf{A}} = \mathbf{U}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are from the singular-value decomposition of $(\mathbf{Y} - \mathbf{Z}\mathbf{D})^T \mathbf{X}\mathbf{B} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ [Ruan (2019)].

Second, update $\mathbf{B}$ for fixed $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{D}}$. Following Chen and Huang (2012), we show in Appendix A that

$$\left\| (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T) \right\|^2 = \left\| (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B} \right\|^2 + K, \tag{2.4.5}$$

for a term K that is constant in $\mathbf{B}$. Inserting the above into equation (2.4.3) implies that to update $\mathbf{B}$ for fixed $(\tilde{\mathbf{A}}, \tilde{\mathbf{D}})$ we solve:

$$\min_{B} \left\| (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B} \right\|^2 + \lambda_1 \sum_{j=1}^{p} \left\| \mathbf{B}^j \right\|_2 \tag{2.4.6}$$

which is a standard convex optimization problem with unknown variable $\mathbf{B}$. We are searching for a penalized matrix $\mathbf{B}$ that trades off small differences between $(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}}$ and $\mathbf{X}\mathbf{B}$, with small values of the coefficients.

Lastly, we update $\tilde{\mathbf{D}}$ for fixed ($\tilde{\mathbf{A}}$ and $\mathbf{B}$). Equation in (2.4.3) can be written as

$$\begin{aligned} &\min_{\tilde{D}} \left\| (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T \right\|^2 \\ = &\min_{\tilde{D}} \left\| (\tilde{\mathbf{Y}} - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T) - \mathbf{Z}\tilde{\mathbf{D}} \right\|^2 \end{aligned} \tag{2.4.7}$$

which is an ordinary least-squares regression of $\mathbf{Y} - \mathbf{X}\mathbf{B}\mathbf{A}^T$ on $\mathbf{Z}$ with unknown variable $\tilde{\mathbf{D}}$. We are looking for a matrix $\tilde{\mathbf{D}}$ that maximizes the linear association between $(\tilde{\mathbf{Y}} - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)$ and $\mathbf{Z}$.

To summarize, the Cov-Con-SRRR algorithm estimates the precision matrix $\mathbf{\Omega}$ for fixed ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$) by applying the GLASSO algorithm, and estimates ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$) for fixed $\mathbf{\Omega}$ using a block-wise coordinate decent method that can be viewed as an extension of the standard approach to solving the SRRR problem. The algorithm is shown in Algorithm 1, below.

Algorithm 1: Sparse reduced-rank regression with covariance estimation and confounders adjustment

```
Input: X, Y, Z, $\lambda_1, \lambda_2$, r
Output: A, B, D, $\mathbf{\Omega}$
initialization;
while objective function in (2.3.5) not converged do
    For fixed ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$), estimate $\mathbf{\Omega}$ with objective function (2.4.1) via GLASSO algorithm;
    For fixed $\mathbf{\Omega}$, estimate $(\tilde{\mathbf{A}}, \mathbf{B}, \tilde{\mathbf{D}})$ defined by the standard SRRR problem (2.3.5), where $\tilde{\mathbf{A}} = \mathbf{\Omega}^{\frac{1}{2}}\mathbf{A}$ and $\tilde{\mathbf{D}} = \mathbf{D}\mathbf{\Omega}^{\frac{1}{2}}$. Set $\mathbf{A} = \mathbf{\Omega}^{-\frac{1}{2}}\tilde{\mathbf{A}}$ and $\mathbf{D} = \tilde{\mathbf{D}}\mathbf{\Omega}^{-\frac{1}{2}}$;
    while objective function in (2.4.2) not converged do
        For fixed $(\mathbf{B}, \tilde{\mathbf{D}})$, update $\tilde{\mathbf{A}}$ defined by the orthogonal Procrustes problem (2.4.4);
        For fixed $(\tilde{\mathbf{A}}, \tilde{\mathbf{D}})$, update $\mathbf{B}$ defined by the SRRR problem (2.4.6);
        For fixed $(\tilde{\mathbf{A}}, \mathbf{B})$, update $\tilde{\mathbf{D}}$ defined by the OLS problem (2.4.7);
    end
end
```

Chapter 3

Results

In this section, we set up two studies to test our Cov-Con-SRRR algorithm. Both studies include the tuning parameter determination, estimation of the matrices of coefficients and the precision matrix, variable importance, and comparisons between the Cov-Con-SRRR and the Cov-SRRR on adjusted variables. Ruan (2019) found that as rank increases, the computing cost associated with fitting the model also increases. Thus, to limit computation, we decided to choose rank $r = 1$.

3.1 Genetic Data

The genetic data we used in this section is from ADNI-1 (http://adni.loni.usc.edu/study-design/). We use the subset of 200 cognitively-normal (CN) subjects. We applied PCA on SNP genotypes obtained from all 33 genes and used the top 10 PCs as our confounding variable $\mathbf{Z}$. The predictor variables $\mathbf{X}$ include 397 SNPs from the gene *NEDD9* and the response variables $\mathbf{Y}$ are 56 MRI measurements. For the predictor matrix, we removed SNPs with more than 5% missing values, and then subjects with more than 5% missing values. Unfortunately this left only 64 subjects and 360 SNPs for our analysis.

3.1.1 Imaging Phenotype Data

The response variables are 56 measures of volumes or cortical thicknesses from 28 brain regions of interest (ROIs) and both hemispheres. These measures were from baseline MRI scans in the ADNI-1 study and were adjusted for covariates such as age, gender, education level, handedness and baseline intracranial volume. Since handedness and baseline intracranial volume could have a genetic basis and be confounders, we might need to directly adjust those two covariates in future work. The following table gives a brief description of each ROI.

	Phenotype ID	**Measurement**	**Cerebral region**
1	AmygVol	Volume	Amygdala
2	CerebCtx	Volume	Cerebral cortex
3	CerebWM	Volume	Cerebral white matter
4	HippVol	Volume	Hippocampus
5	InfLatVent	Volume	Inferior lateral ventricle
6	LatVent	Volume	Lateral ventricle
7	EntCtx	Thickness	Entorhinal cortex
8	Fusiform	Thickness	Fusiform gyrus
9	InfParietal	Thickness	Inferior parietal gyrus
10	InfTemporal	Thickness	Inferior temporal gyrus
11	Midtemporal	Thickness	Middle temporal gyrus
12	Parahipp	Thickness	Parahippocampal gyrus
13	PostCing	Thickness	Posterior cingulate
14	Postcentral	Thickness	Postcentral gyrus
15	Precentral	Thickness	Precentral gyrus
16	Precuneus	Thickness	Precuneus
17	SupFrontal	Thickness	Superior frontal gyrus
18	SupParietal	Thickness	Superior parietal gyrus
19	SupTemporal	Thickness	Superiot temporal gyrus
20	Supramarg	Thickness	Supramarginal gyrus
21	TemporalPole	Thickness	Temporal pole
22	MeanCing	Mean thickness	Caudal anterior cingulate, isthmus cingulate, posterior cingulate, and rostral anterior cingulate

23	MeanFront	Mean thickness	Caudal midfrontal, rostral midfrontal, superior frontal, lateral orbitofrontal, and medial orbitofrontal gyri and frontal pole
24	MeanLatTemp	Mean thickness	Inferior temporal, middle temporal, and superior temporal gyri
25	MeanMedTemp	Mran thickness	Fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole
26	MeanPar	Mean thickness	Inferior and superior parietal gyri, supramarginal gyrus, and precuneus
27	MeanSensMotor	Mean thickness	Precentral and postcentral gyri
28	MeanTemp	Mean thickness	Inferior temporal, middle temporal, superior temporal, fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole

Table 3.1: Brain Imaging Phenotypes: IDs and descriptions of 28 brain regions from each hemisphere, taken from Table 2.1 of Szefer (2014). Baseline structural MRI measurements of a total of 56 (=28 $\times$ 2) regions from left and right hemisphere were estimated

3.1.2 Tuning Process

Considering the limited sample size, we set up a 4-fold cross-validation (CV) along with a hyper grid of λ_1 and λ_2 to determine the optimal values of tuning parameters. Thus, for the CV, we split the dataset into four equal folds and build the model based on three folds. Then predict the response variable for the hold-out fold and compute the mean square prediction error (MSPE). Repeat the above process for each fold and calculate the average MSPE for each combination of λ_1 and λ_2. The one with minimum average MSPE will be the final optimal values of λ_1 and λ_2. Notice that λ_1 is the tuning parameter for the coefficient matrix $\hat{\mathbf{B}}$ and λ_2 is the tuning parameter for the precision matrix $\hat{\boldsymbol{\Omega}}$.

We set up a 7×7 grid of λ_1 and λ_2 for tuning and both take the values from the set $\{0.01, 0.1, 0.5, 1, 5, 50, 100\}$. Results are shown in Table 3.2 and Figure 3.1; we see a minimum MSPE value of 1.206 at $\lambda_1 = 0.5$ and $\lambda_2 = 0.1$.

λ_1 \ λ_2	0.01	0.1	0.5	1	5	50	100
0.01	4.227	1.461	2.328	4.723	3.172	3.158	3.16
0.1	3.213	1.273	1.594	1.724	1.506	1.487	1.485
0.5	1.213	1.206	1.226	1.336	1.259	1.258	1.258
1	1.213	1.215	1.207	1.241	1.213	1.213	1.213
5	1.213	1.213	1.213	1.213	1.213	1.213	1.213
50	1.213	1.213	1.213	1.213	1.213	1.213	1.213
100	1.213	1.213	1.213	1.213	1.213	1.213	1.213

Table 3.2: Genetic data: summary of MSPE

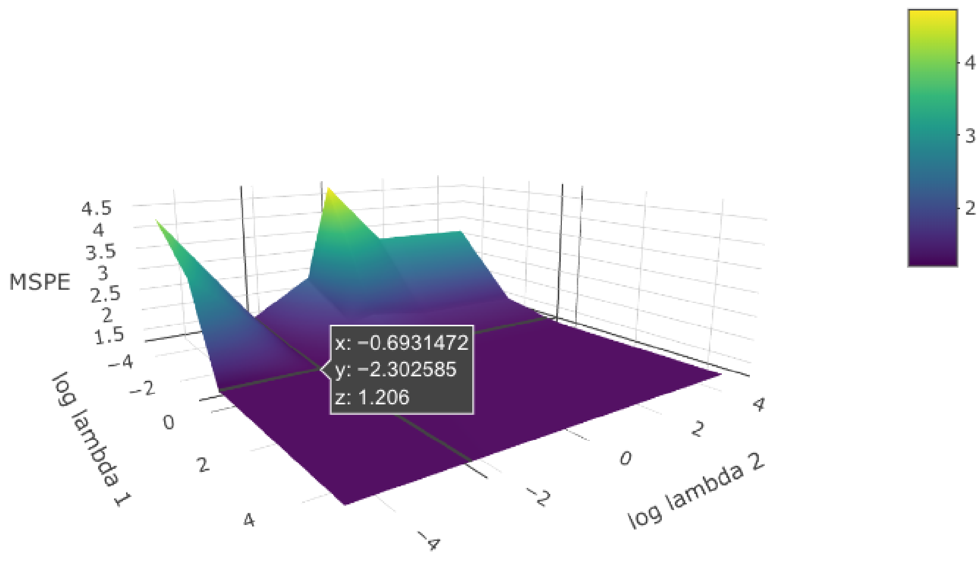

Figure 3.1: Genetic Data: 3D surface of MSPEs; x-axis and y-axis are in log scale

3.1.3 Estimated Coefficients

We set $\lambda_1 = 0.5$ and $\lambda_2 = 0.1$ based on the results above. With these values of the tuning parameters, the estimated coefficient matrix $\hat{\mathbf{B}}$ has only one non-zero row, corresponding to SNP 34, *rs149860773*, which suggests that variation in the SNP *rs149860773* in gene *NEDD9* is associated with differences in the brain ROIs.

Table 3.3 below shows the top 10 absolute value coefficients in the estimated $\hat{\mathbf{A}}$ and the brain ROIs that they correspond to. SupParietal, Precuneus, InfParietal, Postcentral, MeanPar, Precentral, MeanSensMotor, Supramarg in the left-hemisphere and Precuneus, SupParietal in the right-hemisphere are the top 10 ROIs that associated with the SNP 34.

ROIs	hemisphere	measurements	estimated $\mathbf{A}$
SupParietal	Left	Thickness	0.7
Precuneus	Left	Thickness	0.69
InfParietal	Left	Thickness	0.68
Postcentral	Left	Thickness	0.66
MeanPar	Left	Mean thickness	0.63
Precentral	Left	Thickness	0.57
MeanSensMotor	Left	Mean thickness	0.54
Precuneus	Right	Thickness	0.54
SupParietal	Right	Thickness	0.53
Supramarg	Left	Thickness	0.52

Table 3.3: Top 10 coefficients in the estimated $\mathbf{A}$ and their corresponding brain ROIs

3.1.4 Estimated Precision Matrix

The estimated precision matrix $\hat{\mathbf{\Omega}}$ has 1143 out of 1540 nonzero off-diagonal elements. A heatmap of the estimated values is shown in Figure 3.2. The noticeable banding of negative values on the off-diagonal (e.g., between response variables 1 and 29, 2 and 30, 3 and 31, etc.) correspond to positive covariances between left- and right-hemisphere measures of the same brain ROIs.

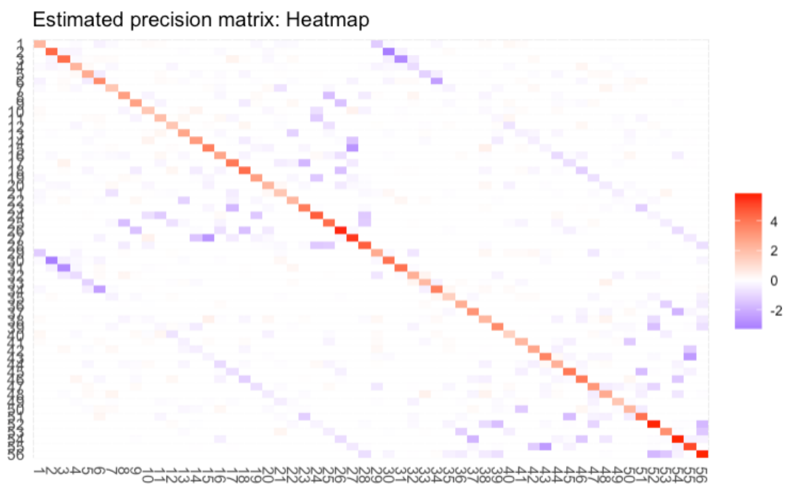

Figure 3.2: Estimated precision matrix: Heatmap

3.1.5 Variable Importance

We also generate 100 bootstrap samples to assess the variable importance for SNP 34 (*rs149860773*). We fit 100 Cov-Con-SRRR models with $\lambda_1 = 0.5$ and $\lambda_2 = 0.1$. Seven of them failed to converge, while the remaining 93 models returned valid coefficient matrices $\hat{\mathbf{B}}$. Recall from equation (2.3.2) that setting the row of $\mathbf{B}$ to zero is equivalent to excluding the corresponding predictor variable $\mathbf{X}$. We find that 19 out of 93 models select SNP 34, giving an importance probability 20.43%.

3.1.6 Confounder adjustment

Here we compare the results of our Cov-Con-SRRR to Cov-SRRR with adjusted response and predictor variables. The adjustments to the response variables $\mathbf{Y}$ and predictor variables $\mathbf{X}$ are as follows. We perform a regression of $\mathbf{Y}$ on $\mathbf{Z}$ and calculate the residuals $\mathbf{e}_Y$ from this regression. These contain the information about the part of $\mathbf{Y}$ not explained by $\mathbf{Z}$. We then perform a regression of $\mathbf{X}$ on $\mathbf{Z}$, and calculate the residuals $\mathbf{e}_X$ from this regression. These represent the part of $\mathbf{X}$ not explained by $\mathbf{Z}$.

With the data from ADNI-1 we find that the Cov-SRRR of $\mathbf{e}_Y$ on $\mathbf{e}_X$ gives an estimated $\mathbf{B}$ with three non-zero rows, corresponding to SNPs 34, 314 and 349.

The Cov-SRRR of $\mathbf{e}_Y$ on $\mathbf{e}_X$ gives an estimated $\mathbf{\Omega}$ that has similar pattern of banding compared to the results of Cov-Con-SRRR. A heatmap of the estimated values is shown in figure 3.3.

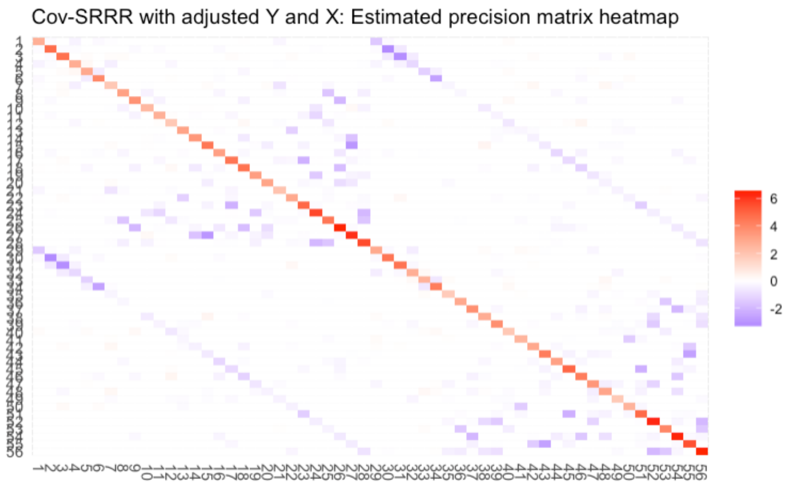

Figure 3.3: Cov-SRRR on residuals: Estimated precision matrix heatmap

3.2 Simulated Data Study

Following Chen and Huang (2016), we generate some simulated data to test the Cov-Con-SRRR algorithm with $q = 4$ response variables. To ensure enough sample size and high dimensional data, we set $n = 400$ subjects and $p = 1000$ predictor variables. To limit computations, we again decided to choose rank $r = 1$.

3.2.1 Simulated Data

We suppose a source population that consists of four different sub-populations. From each sub-population we sample 100 subjects and 10000 genome-wide SNPs markers for each subject. SNP genotypes are simulated independently from population-specific minor allele frequencies (MAFs) from the CDX (Chinese Dai in Xishuang-banna, China), CHB (Han Chinese in Bejing, China), FIN (Iberian population in Spain), and IBS (Finnish in Finland) populations in the 1000Genomes database.

We apply PCA to the simulated genome-wide genotypes. The following plots show that the top 3 PCs separate the four sub-populations. We choose the top 10 PCs as our confounding variable $\mathbf{Z}$, so that $\mathbf{Z}$ is a 400×10 matrix.

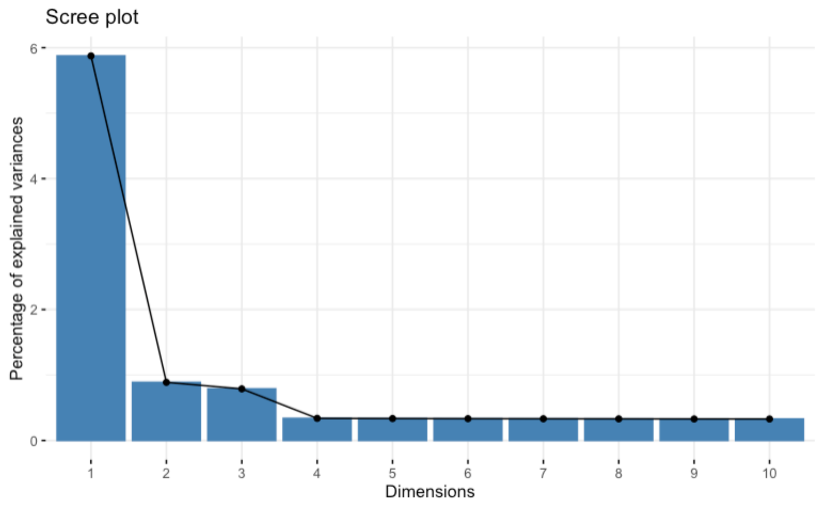

Figure 3.4: Simulated data:PCA scree plot

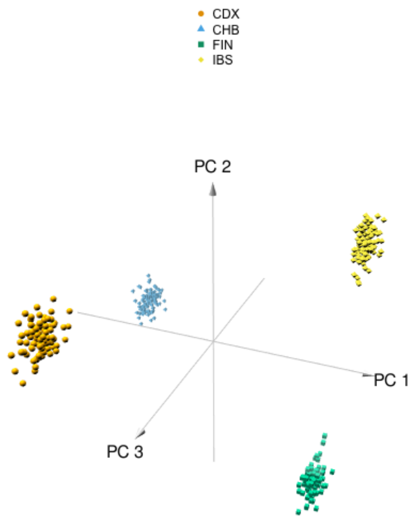

Figure 3.5: Simulated data:3D PCA plot

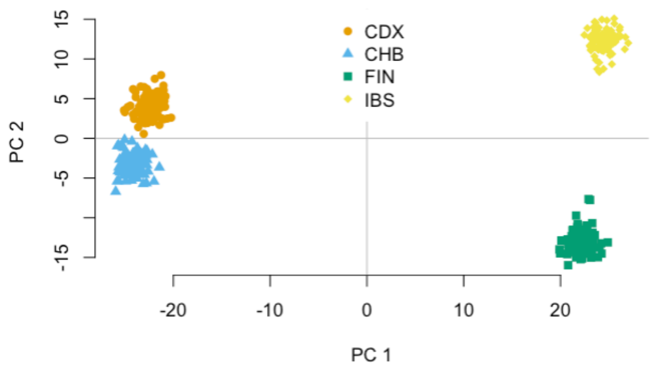

Figure 3.6: Simulated data:2D PCA plot

We next suppose that the target gene $\mathbf{X}$ consists of 1000 SNPs. Following [Chen and Huang (2016)], we generate the error matrix $\boldsymbol{E}_{400\times 4}$ from $N(0, \sigma^2\boldsymbol{\Sigma}_e)$, where σ and $\boldsymbol{\Sigma}_e$ are 0.1 and $\begin{bmatrix} 1 & 0.9 & 0 & 0 \\ 0.9 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.9 \\ 0 & 0 & 0.9 & 1 \end{bmatrix}$, respectively. For the coefficient matrix $\mathbf{B}_{1000\times 1}$, the first five column were set to 1 and the remaining 1000 - 5 columns were set to zero. The elements of the matrix $\mathbf{A}_{4\times 1}$ and $\mathbf{D}_{10\times 4}$ are set to 1.

We randomly select 1000 markers from the 10000 genome-wide SNP markers, such that $\mathbf{X}$ is a 400×1000 predictor matrix. The simulated response variable can be written as:

$$\mathbf{Y} = \mathbf{XBA}^T + \mathbf{ZD} + \mathbf{E}$$

where $\mathbf{Y}$ is a 400×4 matrix.

3.2.2 Tuning Process

Using a 5 fold-CV strategy, we set up a 8×8 grid of λ_1 and λ_2 for tuning and both take the values from the set $\{0.001, 0.01, 0.1, 0.5, 1, 5, 50, 100\}$.

Results are shown in Table 3.4 and Figure 3.7; we see a minimum MSPE value of 0.134 at $\lambda_1 = 0.1$ and $\lambda_2 = 5$.

λ_1 \ λ_2	0.001	0.01	0.1	0.5	1	5	50	100
0.001	7.801	1.028	0.554	0.274	0.267	0.179	0.173	0.173
0.01	14.648	0.688	0.170	0.259	0.238	0.172	0.174	0.167
0.1	4.497	3.013	1.001	0.159	0.136	0.134	0.134	0.236
0.5	3.531	3.416	2.508	0.219	0.174	0.182	0.182	0.165
1	4.381	4.359	4.114	1.668	0.499	0.476	0.476	0.476
5	4.390	4.390	4.390	4.390	4.390	4.390	4.390	4.390
50	4.390	4.390	4.390	4.390	4.390	4.390	4.390	4.390
100	4.390	4.390	4.390	4.390	4.390	4.390	4.390	4.390

Table 3.4: Simulated data: summary of MSPE

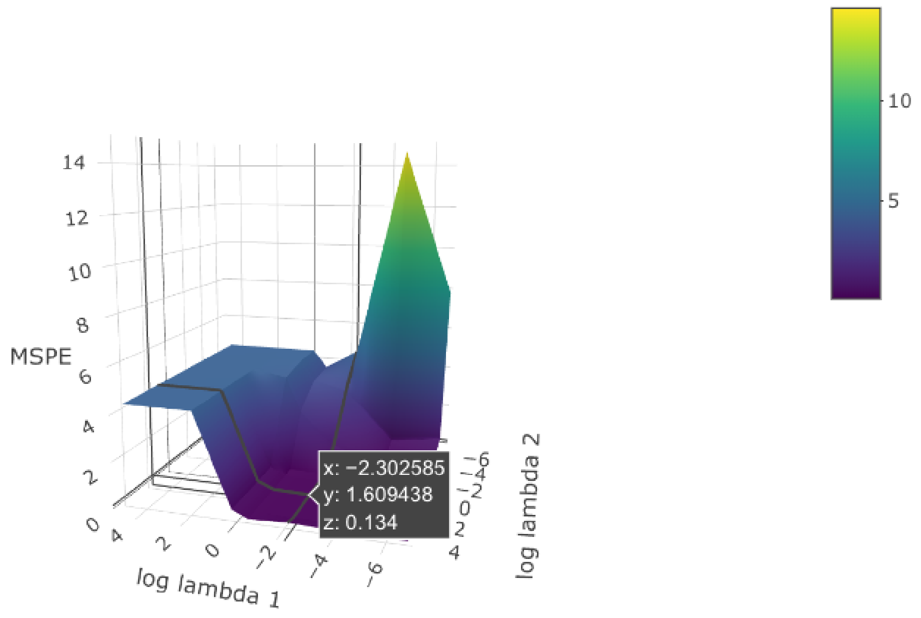

Figure 3.7: Simulated Data: 3D surface of MSPEs; x-axis and y-axis are in log scale

3.2.3 Estimated Coefficients

Based on the CV results from Section 3.2.2, we set $\lambda_1 = 0.1$ and $\lambda_2 = 5$. The estimated coefficient matrix $\hat{\mathbf{B}}$ has 105 non-zero rows including the 5 truly associated SNPs in $\mathbf{B}$, which means that the 105 non-zero SNP markers are considered to be important and associated with the response variable $\mathbf{Y}$. This is in contrast to the five non-zero rows in the matrix $\mathbf{B}$ used to simulate the data.

We also generate 100 bootstrap samples to assess the variable importance of the 5 truly associated SNPs. We find that all 100 Cov-Con-SRRR models select the 5 truly associated SNPs, giving an importance probability 100%.

Table 3.5 below shows the estimated coefficients in $\hat{\mathbf{A}}$ and the response features that they correspond to. Recall from Chapter 2 that $\mathbf{A}$ is only unique up to an $r \times r$ orthonormal matrix. In the case of $r = 1$ the orthonormal matrices are 1 and -1; i.e., the results are unique up to a sign change. The elements of the matrix $\mathbf{A}$ used to simulate the data are all equal to 1. The elements of $\hat{\mathbf{A}}$ are all smaller in magnitude.

Y	estimated $\mathbf{A}$
feature 1	-0.1709
feature 2	-0.1661
feature 3	-0.1684
feature 4	-0.1619

Table 3.5: Coefficients in the estimated $\mathbf{A}$ and their corresponding response features

Since the estimated $\mathbf{B}$ coefficients could compensate for the smaller $\hat{\mathbf{A}}$, we decide to compare the estimated $\mathbf{BA}^T$ to true $\mathbf{BA}^T$. The mean squared difference between the estimate $\hat{\mathbf{C}} = \hat{\mathbf{B}}\hat{\mathbf{A}}^T$ and the matrix $\mathbf{C} = \mathbf{BA}^T$ used to simulated the data is:

$$diff = \frac{1}{n}\sum(\mathbf{BA}^T - \hat{\mathbf{B}}\hat{\mathbf{A}}^T)^2 = 2.7026 \times 10^{-5}$$

The mean squared difference between the matrix $\mathbf{D}$ used to simulate the data and its estimate $\hat{\mathbf{D}}$ is:

$$diff = \frac{1}{n}\sum(\mathbf{D} - \hat{\mathbf{D}})^2 = 3.2675 \times 10^{-6}$$

3.2.4 Estimated Precision Matrix

The estimated precision matrix $\hat{\mathbf{\Omega}}$ is diagonal. Off-diagonal elements equal to zero means that all four response variables are conditionally independent from each other, given all other variables. This is in contrast to the $\mathbf{\Omega}$ used to simulate the data, which included correlation between the $\mathbf{Y}_i$'s. Thus, the penalization in estimation of the precision matrix appears to over-shrink the estimates to zero.

3.2.5 Confounder adjustment

Here we compare the results of our Cov-Con-SRRR to Cov-SRRR on adjusted response and predictor variables, where the adjusted variable are residuals from regression on the confounding variables. Following the same procedure in the genetic study, we perform Cov-SRRR on the residuals $\mathbf{e}_Y$ on $\mathbf{e}_X$ from regressions on the confounder variables $\mathbf{Z}$. The resulting estimated $\hat{\mathbf{B}}$ has 117 non-zero rows and the 5 truly associated SNPs were included.

Chapter 4

Conclusions

This project demonstrates an approach we call Cov-Con-SRRR for multiple-response regression that incorporates shrinkage and variable selection of predictor variables and covariances, and direct adjustment for confounder variables. Prior to this approach, the only method for confounder adjustment was to perform Cov-SRRR on residual response and predictor variables, where the residuals are from regressions on the confounder variables.

We applied the algorithm to genetic data that includes 64 cognitively normal subjects from the ADNI-1 study to explore the relationship between 56 brain imaging measures and 360 SNPs from the gene *NEDD9*. To adjust for population structure we included the top 10 PCs calculated from genome-wide SNPs as confounders. Tuning parameters were determined by 4-fold CV over a 7×7 hyper grid of λ_1 and λ_2. Our Cov-Con-SRRR results suggest that a single SNP, *rs149860773*, in *NEDD9* is associated with the phenotypes. However, the variable importance of *rs149860773* was a relatively modest 20.43%. The estimated precision matrix $\hat{\mathbf{\Omega}}$ is non-diagonal, with correlations between left- and right-hemisphere measures of the same brain region. Repeating the analysis as a standard Cov-SRRR on residual phenotypes and predictors, adjusted for the confounders, suggested three SNPs with non-zero coefficients.

Choi et al. (2019) also studied associations between the 56 brain phenotypes and SNPs in *NEDD9*. They found that the SNP *rs16871157* is associated with measures of cortical thickness. We were not able to replicate this finding because *rs16871157* was removed during data cleaning. This, and the small sample size of 64 were consequences of excessive missing genetic data, despite efforts at data imputation. Thus, an important area for future work is to revisit the imputation process in an attempt to include more subjects and more SNPs in the analysis.

We also tested the algorithm on simulated data for 400 subjects, where the 400 individuals were generated from four different populations. After selection of the tuning parameters we found 105 SNPs with non-zero coefficients. However, the true coefficient matrix $\mathbf{B}$ has only 5 non-zero rows and all of them were included in the 105 SNPs. The 100 bootstrap samples also show that the variable importance of the 5 truly associated SNPs are 100%. Thus, shrinkage on the matrix $\mathbf{B}$ did not achieve the same degree of sparsity as the truth. Unlike the results in the genetic study, the estimated precision matrix $\hat{\boldsymbol{\Omega}}$ in this simulated data study is diagonal, even though the true precision matrix is non-diagonal. Thus, for estimation of the precision matrix, penalization leads to over-shrinkage. Repeating the analysis as a standard Cov-SRRR on residual phenotypes and predictors, adjusted for the confounders, suggested 117 SNPs with non-zero coefficients and the 5 truly associated SNPs were included.

We have to assign initial values for $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ before applying the algorithm. In this project, we choose to use the OLS solution of $\mathbf{D}$ in (2.1.3) as the initial start, and there is an R function called *rrr()* that fits a reduced-rank regression and returns the estimated coefficients of $\mathbf{A}$ and $\mathbf{B}$. It is also possible to randomly assign the value of $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$. But under this situation, most of the time our algorithm fails to converge as the initial points might be far away from the true value.

Cov-Con-SRRR incorporates confounding variables directly into the model and model-fitting procedure, which could offer advantages over *ad hoc* adjustment for confounders through Cov-SRRR analysis of residual phenotype and predictor variables. Cov-Con-SRRR incurs an extra computational cost over Cov-SRRR on residuals because estimation of the confounder effects is added to the block-wise coordinate descent of the Cov-Con-SRRR algorithm. Our analyses of real and simulated data show that Cov-Con-SRRR and Cov-SRRR on residuals give *different* results, but further study is required to assess whether Cov-Con-SRRR is *better*, and whether any improvements are worth the extra computational cost. Computational costs for both Cov-Con-SRRR and Cov-SRRR are compounded by the need to run the algorithms repeatedly to determine values of the shrinkage parameters. For example, with the ADNI-1 data, it takes about eight hours to tune the regularization parameters λ_1 and λ_2. The computing cost associated with fitting the model also increase as the rank r

increases. Due to computational limits, we only examined the algorithm for $r = 1$. Thus, an area for future work is to develop more computationally efficient methods for selecting the shrinkage and rank parameters.

Bibliography

Chen, L. and Huang, J. Z. (2012), 'Sparse reduced-rank regression for simultaneous dimension reduction and variable selection', *Journal of the American Statistical Association* **107**(500), 1533–1545.

Chen, L. and Huang, J. Z. (2016), 'Sparse reduced-rank regression with covariance estimation', *Statistics and Computing* **26**(1-2), 461–470.

Choi, J., Lu, D., Beg, M. F., Graham, J., McNeney, B., (ADNI, A. D. N. I. et al. (2019), 'The contribution plot: Decomposition and graphical display of the rv coefficient, with application to genetic and brain imaging biomarkers of alzheimer's disease', *Human heredity* **84**(2), 59–72.

Halani, K. A. (2016), 'Sparse multivariate reduced-rank regression with covariance estimation'.

Izenman, A. J. (1975), 'Reduced-rank regression for the multivariate linear model', *Journal of multivariate analysis* **5**(2), 248–264.

Li, Y., Grupe, A., Rowland, C., Holmans, P., Segurado, R., Abraham, R., Jones, L., Catanese, J., Ross, D., Mayo, K. et al. (2008), 'Evidence that common variation in nedd9 is associated with susceptibility to late-onset alzheimer's and parkinson's disease', *Human molecular genetics* **17**(5), 759–767.

Mazumder, R. and Hastie, T. (2012), 'The graphical lasso: New insights and alternatives', *Electronic journal of statistics* **6**, 2125.

Menozzi, P., Piazza, A. and Cavalli-Sforza, L. (1978), 'Synthetic maps of human gene frequencies in europeans', *Science* **201**(4358), 786–792.

Nutini, J., Laradji, I. and Schmidt, M. (2017), 'Let's make block coordinate descent go fast: Faster greedy rules, message-passing, active-set complexity, and superlinear convergence', *arXiv preprint arXiv:1712.08859* .

Ruan, H. (2019), 'Covariance-adjusted, sparse, reduced-rank regression with application to imaging-genetics data'.

Szefer, E. (2014), 'Joint analysis of imaging and genomic data to identify associations related to cognitive impairment'.

Velu, R. and Reinsel, G. C. (2013), *Multivariate reduced-rank regression: theory and applications*, Vol. 136, Springer Science & Business Media.

Weisberg, S. (2013), *Applied linear regression*, John Wiley & Sons.

Appendix A

Appendix: Proof of (2.4.5)

Recall, we have the Cov-Con-SRRR model:

$$\begin{aligned}\mathbf{Y}_{n\times q} &= \mathbf{X}_{n\times p}\mathbf{C}_{p\times q} + \mathbf{Z}_{n\times 10}\mathbf{D}_{10\times q} + \mathbf{E}_{n\times q}\\ &= \mathbf{X}_{n\times p}\mathbf{B}_{p\times r}\mathbf{A}^T_{r\times q} + \mathbf{Z}_{n\times 10}\mathbf{D}_{10\times q} + \mathbf{E}_{n\times q}\end{aligned}$$

where $rank(\mathbf{C}) = r, r \leq min(p,q)$. And for fixed $\mathbf{\Omega}$, the optimization problem in (2.4.3) for solving $(\tilde{\mathbf{A}}, \mathbf{B}, \tilde{\mathbf{D}})$ is:

$$\begin{aligned}\min_{\tilde{A},B,\tilde{D}} \quad & \frac{1}{n}\left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\right\|^2 + \lambda_1 \sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|^2\\ & s.t. \tilde{\mathbf{A}}^T\tilde{\mathbf{A}} = \mathbf{I}_r\end{aligned}$$

where $\tilde{\mathbf{A}} = \mathbf{\Omega}^{\frac{1}{2}}\mathbf{A}$, $\tilde{\mathbf{D}} = \mathbf{D}\mathbf{\Omega}^{\frac{1}{2}}$ and $\tilde{\mathbf{Y}} = \mathbf{Y}\mathbf{\Omega}^{\frac{1}{2}}$. We can conclude that the matrix $\tilde{\mathbf{A}}$ has orthonormal columns as $\tilde{\mathbf{A}}^T\tilde{\mathbf{A}} = \mathbf{I}_r$.

Now consider the case that $C = B^*_{p\times(q-r)}{A^*_{(q-r)\times q}}^T$ where rank($\mathbf{C}$) = q - r, and q - r $\leq min(p,q)$. Then, the optimization problem in (2.4.4) for solving $\mathbf{A}^*$ can be written as:

$$\begin{aligned}&\min_{\tilde{A^*}}\left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}^*\tilde{\mathbf{A}^*}^T\right\|^2,\\ &s.t. \tilde{\mathbf{A}^*}^T\tilde{\mathbf{A}^*} = \mathbf{I}_{q-r}\end{aligned} \tag{A.0.1}$$

Thus, matrix$\tilde{\mathbf{A}}^*_{q\times(q-r)}$ has orthonormal columns as $(\tilde{\mathbf{A}}^*)^T\tilde{\mathbf{A}}^* = \mathbf{I}_{q-r}$. Combine matrix $\tilde{\mathbf{A}}_{q\times r}$ and $\tilde{\mathbf{A}}^*{}_{q\times(q-r)}$ together, matrix $(\tilde{\mathbf{A}}, \tilde{\mathbf{A}}^*)_{q\times q}$ is a orthogonal matrix as its columns are orthonormal vector.

Rewrite the objective function in (A.1), we have:

$$\begin{aligned}\left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T\right\|^2 &= \left\|((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)(\tilde{\mathbf{A}}, \tilde{\mathbf{A}}^*)\right\|^2 \qquad (A.0.2)\\ &= \left\|((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}} \ \mid \ ((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}}^*\right\|^2 \qquad (A.0.3)\end{aligned}$$

we partitioned the matrix in (A.2) into two parts, for the left hand side of the matrix, $((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}}$, we have:

$$((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}} = (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T\tilde{\mathbf{A}} = (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B} \qquad (A.0.4)$$

For the right hand side of the matrix, $((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}}^*$, we have:

$$((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}}^* = (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}}^* - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T\tilde{\mathbf{A}}^* = (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}}^* \qquad (A.0.5)$$

Proof. (A.5)

Matrix $(\tilde{\mathbf{A}}, \tilde{\mathbf{A}}^*)_{q\times q} = \begin{bmatrix} \tilde{A}_{11} & \dots & \tilde{A}_{1r} & \tilde{A}^*_{11} & \dots & \tilde{A}^*_{1(q-r)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{A}_{q1} & \dots & \tilde{A}_{qr} & \tilde{A}^*_{q1} & \dots & \tilde{A}^*_{q(q-r)} \end{bmatrix}$ is a orthogonal matrix with orthonormal column vectors,

Thus:

$$\begin{aligned}\tilde{\mathbf{A}}^T_{r\times q} \cdot \tilde{\mathbf{A}}^*_{q\times(q-r)} &= \begin{bmatrix} \tilde{A}_{11} & \dots & \tilde{A}_{1r} \\ \vdots & \vdots & \vdots \\ \tilde{A}_{q1} & \dots & \tilde{A}_{qr} \end{bmatrix}^T \cdot \begin{bmatrix} \tilde{A}^*_{11} & \dots & \tilde{A}^*_{1(q-r)} \\ \vdots & \vdots & \vdots \\ \tilde{A}^*_{q1} & \dots & \tilde{A}^*_{q(q-r)} \end{bmatrix} \\ &= \begin{bmatrix} \tilde{A}_{11} & \dots & \tilde{A}_{q1} \\ \vdots & \vdots & \vdots \\ \tilde{A}_{1r} & \dots & \tilde{A}_{qr} \end{bmatrix} \cdot \begin{bmatrix} \tilde{A}^*_{11} & \dots & \tilde{A}^*_{1(q-r)} \\ \vdots & \vdots & \vdots \\ \tilde{A}^*_{q1} & \dots & \tilde{A}^*_{q(q-r)} \end{bmatrix} \\ &= 0\end{aligned}$$

$\square$

Combine results (A.4) and (A.5), we have:

$$\begin{aligned}
&\left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T\right\|^2 \\
=&\left\|((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)(\tilde{\mathbf{A}}, \tilde{\mathbf{A}}^*)\right\|^2 \\
=&\left\|[((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}} \ \mid \ ((\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\tilde{\mathbf{A}}^*]\right\|^2 \\
=&\left\|[(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B} \ \mid \ (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}}^*]\right\|^2
\end{aligned} \tag{A.0.6}$$

Let $\mathbf{W}$ denotes $(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B}$ and let $\mathbf{V}$ denotes $(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}}^*$. Hence, (A.6) can be written as:

$$\begin{aligned}
&\left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T\right\|^2 = \left\|\begin{bmatrix}\mathbf{W} & \mathbf{V}\end{bmatrix}\right\|^2 = \sqrt{tr(\begin{bmatrix}\mathbf{W} & \mathbf{V}\end{bmatrix}^T \begin{bmatrix}\mathbf{W} & \mathbf{V}\end{bmatrix})} \\
&= \sqrt{tr(\mathbf{W}^T\mathbf{W}) + tr(\mathbf{V}^T\mathbf{V})} \leq \sqrt{tr(\mathbf{W}^T\mathbf{W})} + \sqrt{tr(\mathbf{V}^T\mathbf{V})} = \|\mathbf{W}\|^2 + \|\mathbf{V}\|^2
\end{aligned} \tag{A.0.7}$$

Thus, updating $\mathbf{B}$ for fixed $\mathbf{\Omega}, \mathbf{A}$ and $\mathbf{D}$, we have to solve the following minimization problem:

$$\begin{aligned}
&\min_B \ \frac{1}{n}\left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}}) - \mathbf{X}\mathbf{B}\tilde{\mathbf{A}}^T)\right\|^2 + \lambda_1 \sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2 \\
&= \min_B \ \left\|[(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B} \ \mid \ (\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}}^*]\right\|_2 + \lambda_1 \sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2 \\
&\leq \min_B \ \left[\left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B}\right\|^2 + \left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}}^*]\right\|^2\right] + \lambda_1 \sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2
\end{aligned}$$

Overall, the optimization problem for solving $\mathbf{B}$ for fixed $\mathbf{\Omega}, \mathbf{A}$ and $\mathbf{D}$, up to a constant, can be written as:

$$\min_B \ \left\|(\tilde{\mathbf{Y}} - \mathbf{Z}\tilde{\mathbf{D}})\tilde{\mathbf{A}} - \mathbf{X}\mathbf{B}\right\|^2 + \lambda_1 \sum_{j=1}^{p}\left\|\mathbf{B}^j\right\|_2$$

Appendix B

Appendix: Functions

```
#------------------------------------------------------------
#Chen and Huang (2016) algorithm 1 as coded by Khalif
#updated by professor Jinko Graham, Simon Fraser University
#updated by Gloria, for adjusting the confounding variable
#------------------------------------------------------------
Algorithm_1 = function(Y,X,Z,D,l1,l2,r,max_iter=1000,
                        max_best=5, verbose=FALSE){
  Y_star <- Y - Z%*%D

  # Initialize O as a diagonal matrix of inverse variances
  Yvariances <- apply(Y_star,2,var); O = diag(1/Yvariances)

  #Initialize A and B as a SRRR assuming independent errors
  # using the srrr() function from the package rrpack.
  #ab<-rrpack::srrr(Y_star,X,nrank=r,modstr=list(lamA=l2))
  # returns SVD of C=BA^T
  #A<- ab$V; B <- ab$U %*% ab$D

  #Initialize A and B as a RRR
  ab<-rrr::rrr(X, Y_star, type = "identity", rank=1, k=.1)
  A <- as.matrix(ab$A, ncol=1)
  B <- t(as.matrix(ab$B))

  #Initialize: randomly generate A and B
```

```
#A <- matrix(data = rnorm(ncol(Y), mean = 5, sd=1),
                              # ncol = 1)
#B <- matrix(data = rnorm(ncol(X)), ncol = 1)

# - Initialize convergence criterion values
# to arbitrary large numbers
obj_best <- obj_val <- obj_diff <- 1e10
# - Initialize iteration counters to 1
itera <- 1 # main iteration counter
iterb <- 1 # number of iterations
           # with current best value of obj. func.
#---------------------------------------------------------------
# While not converged, iterate between estimation
# of O by GLASSO and (A,B) by SRRR.
while(obj_diff>0.001&&itera<max_iter&&iterb<max_best){

  # 1. GLASSO to estimate O for fixed A,B,D.
  Sigma_R = crossprod(Y_star-X%*%B%*%t(A))/nrow(Y_star)
  O = GLASSO(Sigma_R, l2)

  # 2. SRRR on Ytilde=Y O^{1/2} and X to
  # estimate Atilde, B and D
  Ytilde <- Y %*% chol(O)
  Dtilde <- D %*% chol(O)

  ss<-SRRR(Ytilde,X,r,l1,max_iter,max_best,B,Dtilde,Z,O)
  B <- ss$B
  A <- solve(chol(O)) %*% ss$A
  # Atilde=O^{1/2}A, so A=O^{-1/2}Atilde
  D <- ss$D %*% solve(chol(O))

  #check whether objective function has converged
  #(Chen & Huang (2016), eqn. 4)
  obj_newval = obj_fun_4(Y,X,A,B,O,l1,l2,D,Z)
  obj_diff = abs(obj_val - obj_newval)
```

```
    obj_val = obj_newval
    if(verbose)
    {cat("iteration ",itera,
    " objective func. ",obj_val,"\n")}
    # Check whether objective function has improved
    if(itera > 1 & obj_val < obj_best) {
      obj_best <- obj_val
      iter_best <- itera
      O_best <- O; B_best <- B;
      A_best <- A; D_best <- D
      iterb <- 1
    } else {
      iterb <- iterb + 1
    }
    itera = itera+1
  }
  if(itera==max_iter)
 warning("did not converge after",max_iter,"iterations\n")
  return(list(A = A_best,B = B_best,
   D = D_best, O = O_best,iter=itera, obj=obj_best))
}

# Khalif's function to evaluate objective function 4
# (the negative of the penalized log-likelihood)
# from CH. This is used to monitor convergence.

obj_fun_4 = function(Y,X,A,B,O,l1,l2,D,Z){
  n = nrow(Y)
  Y_star <- Y-Z%*%D
  Sigma_R = crossprod(Y_star-X%*%B%*%t(A))/n
  t1 <- sum(diag(Sigma_R%*%O))
  t2 <- -log(det(O))
  #t1+t2 = negative log-likelihood func.
  t3 <- l2*sum(abs(O[row(O)!=col(O)]))
  #element-wise LASSO penalty on Omega
  t4 <- l1*sum(sqrt(rowSums(B^2)))
  #row-wise sparsity-inducing penalty on B
  return(t1+t2+t3+t4)
}

## GLASSO Update to $\Omega$
GLASSO <- function(Sigma_R, l2){
  library(CVXR)
  p <- nrow(Sigma_R)
  O <- Variable(c(p, p), PSD=TRUE)
  # Semidefinite p*p precision matrix to optimize over
  obj <- matrix_trace(Sigma_R%*%O) - log_det(O) +
           l2*p_norm(O-diag(diag(O)),p=1) #objective func.
  prob <- Problem(Minimize(obj))   # define problem
  result <- solve(prob)            # solve problem
  as.matrix(result$getValue(O))
}

# Implementation of GLASSO using tools from CVXR.
# Note: The penalty term includes the diagonal matrix diag(O)
# whose diagonal elements are those of O.
# In R, you need to call diag twice to get such a matrix,
# once to extract the vector of diagonal terms from O and
# a second time to make a diagonal matrix out of this vector.

## SRRR Update to $A$, $B$ and $D$
SRRR <- function(Ytilde,X,r,l1,max_iter,
                 max_best,B,Dtilde,Z,O) {
  itera <- iterb <- 1
  obj_val <- 1e10; obj_best <- 1e10
  obj_diff <- 10000
  while(abs(obj_diff)>0.001&&
       itera < max_iter&&iterb < max_best){

    # 1. Update A for fixed B, D
```

```
Y <- Ytilde %*% solve(chol(O))
D <- Dtilde %*% solve(chol(O))

Y_star <- Ytilde- Z%*%Dtilde
svd_results = svd(t(Y - Z %*% D)%*%X%*%B,nu=r,nv=r)
A = svd_results$u%*%t(svd_results$v)

# 2. Update B for fixed A, D
# using the optimizer from the CVXR package
n <- nrow(Y_star); p <- ncol(X); q <- ncol(Y)
BB <- Variable(rows=p,cols=r)
# Variable to optimize over

obj <- sum(((Ytilde - Z%*%Dtilde)%*%A - X%*%BB)^2)/n +
  l1*sum(p_norm(BB,p=2,axis=1))
  # SRRR objective function

prob <- Problem(Minimize(obj))  # define problem
result <- solve(prob)           # solve problem
B <- matrix(result$getValue(BB),
            ncol=r,nrow=p) # extract solution


# 3. Update D for fixed A, B
# using the optimizer from the CVXR package
n <- nrow(Y); q <- ncol(Y)
DD <- Variable(rows=nrow(D),cols=q)
# Variable to optimize over, 10 PCs

obj <- sum((Ytilde - X%*%B%*%t(A)-Z%*%DD)^2)/n +
  l1*sum(p_norm(B,p=2,axis=1))
  # SRRR objective function

prob <- Problem(Minimize(obj))  # define problem
result <- solve(prob)           # solve problem
```

```
    Dtilde <- matrix(result$getValue(DD),
                ncol=q,nrow=nrow(D)) # extract solution


    #check whether objective function has converged
    obj_newval = result$value
    obj_diff = (obj_val - obj_newval)
    obj_val = obj_newval
    # Check for improvement, if itera > 1
    if(itera > 1 & obj_val < obj_best) {
      obj_best <- obj_val
      iter_best <- itera
      A_best <- A
      D_best <- Dtilde
      B_best <- B
      iterb <- 1
    } else {
      iterb <- iterb + 1
    }
    itera = itera+1
  }
  return(list(A=A_best,B=B_best, D=D_best))

}

## Cross Validation

* The following is code for
cross validation taken from [ @Halani2016 ].

do_cv = function(params,Y,X,Z,D,num_folds){
l1 = params[1] #tuning for matrix B
l2 = params[2] #tuning for matrix O
if(length(params)==3){
r = params[3]
```

```
}else{
r = 1 #set default for r=1
}
fold_size = nrow(Y)/num_folds

fold_err = numeric(length = num_folds)
for(k in 1:num_folds){
Y_in = Y[-c(((k-1)*fold_size+1):(k*fold_size)),]
X_in = X[-c(((k-1)*fold_size+1):(k*fold_size)),]
Z_in = Z[-c(((k-1)*fold_size+1):(k*fold_size)),]

Y_out = Y[c(((k-1)*fold_size+1):(k*fold_size)),]
X_out = X[c(((k-1)*fold_size+1):(k*fold_size)),]
Z_out = Z[c(((k-1)*fold_size+1):(k*fold_size)),]

res = Algorithm_1(Y_in,X_in,Z_in,D,l1,l2,r)



fold_err[k] = pred_err(Y_out,X_out,Z_out,
           res$A,res$B,res$D)
           #defined pred_err( ) below as mse
message(paste(k,"folds␣done!"))
}
mean_err = mean(fold_err)
print(paste("MSE:",mean_err))
return(mean_err)
}

pred_err = function(Y_out, X_out, Z_out, A, B, D){
Yhat = X_out %*% B %*% t(A) + Z_out%*%D
mse = sum( (Y_out - Yhat)^2) /(length(Yhat))
return(mse)
}
```

Appendix C

Appendix: Genetic study

```
## Imaging data - MRI (from ADNI-1, ADNIGO/2)
mri <- read.csv("MRI_ALL_MERGE.csv")
mri <- mri[,-1]

## Confounders
allchr <- read.table("ADNI1_final_pca.eigenvec")
colnames(allchr)[3:12] <- c("PC1","PC2","PC3","PC4",
                           "PC5","PC6","PC7","PC8","PC9","PC10")
row.names(allchr) <- allchr[,1]
allchr <- allchr[,-c(1,2)]

#eigenvalues
allchr.val <- read.table("ADNI1_final_pca.eigenval")
rownames(allchr.val) <- c("PC1","PC2","PC3","PC4","PC5",
                          "PC6","PC7","PC8","PC9","PC10")

#change the individual ID
pattern3 <- row.names(allchr)
sub_id <- strsplit(pattern3,"_")
sample_id_temp <- NULL
sample_id <- NULL
for (i in 1:691){ #sample size 691 in ADNI1
  sample_id_temp[i] <- as.numeric(sub_id[[i]][3])
}
```

```
sample_id <- paste(sample_id_temp, "bl", sep = "_")
rownames(allchr) <- sample_id

## ADNI1 1 - chr 6; gene NEDD9;
 position: 11,183,298-11,232,668

library(genio)
#chr6 <- read_bim("ADNI1_chr_6.bim")
#ADNI1_NEDD9_list <- chr6[chr6$pos >= 11183298
                          & chr6$pos <= 11232668,]
#write.table(ADNI1_NEDD9_list$id,
"ADNI1_NEDD9_gene_list.txt",row.names = F,
                          sep = "\t", quote = F)
#go back and run PLINK,
#extract gene NEDD9 based on the SNPs list

adni1_nedd9 <- read_plink("ADNI1_gene_NEDD9")
colnames(adni1_nedd9$X) <- adni1_nedd9$fam$fam

#Original data, total 397 features,
#691 subjects (with a lot NA)
nedd9_adni1 <- as.data.frame(t(adni1_nedd9$X))

## CN subject
xx <- read.delim("genosCN.dat", header=T, sep="␣")
yy <- read.delim("phenosCN.dat", header = T, sep="␣")
xx.subject <- row.names(xx)

CN <- nedd9_adni1[rownames(nedd9_adni1)%in% xx.subject,]
#178 subjects, 397 features
#Overall missing varible plot
library(naniar)
vis_miss(CN, show_perc_col = T)

#missing in row or column, percentage
```

```
missing_in_row <- NULL
for (i in 1:nrow(CN)) {
  missing_in_row[i] <- (sum(is.na(CN[i,]))/397)*100
}

missing_in_col <- NULL
for(j in 1:ncol(CN)){
  missing_in_col[j] <- (sum(is.na(CN[,j]))/178)*100
}

row_missing <- data.frame(x=missing_in_row, y=c(1:178))
col_missing <- data.frame(x=missing_in_col, y=c(1:397))

hist(row_missing$x,breaks = 100,
     xlab = "NA␣value␣in␣row␣(%)")
hist(col_missing$x,breaks = 100,
     xlab = "NA␣value␣in␣column␣(%)")

#filter out missing data
#if row has over 5% missing, then delete
#if column has over 5% missing, then delete
filter_missing <- CN[missing_in_row < 5,missing_in_col < 5]
#172 subjects, 360 features
vis_miss(filter_missing)

#clean all rows that contain missing value
na_count <- NULL
filter_nedd9 <- NULL
for (i in 1:nrow(filter_missing)) {
  na_count <- sum(is.na(filter_missing[i,]))
  filter_nedd9 <- c(filter_nedd9,
                    ifelse(na_count <= 0, i, NA))
}

#cleaned data
```

```
#64 subjects, 360 features
nedd9_adni1_filter <- filter_missing[na.omit(filter_nedd9),]

## corresponding Response Variable
#change the individual id
pattern2 <- row.names(nedd9_adni1_filter)
sub_id <- strsplit(pattern2,"_")
sample_id_temp <- NULL
sample_id <- NULL
for (i in 1:nrow(nedd9_adni1_filter)){ #sample size 64
  sample_id_temp[i] <- as.numeric(sub_id[[i]][3])
}
sample_id <- paste(sample_id_temp, "bl", sep = "_")
rownames(nedd9_adni1_filter) <- sample_id

#find the corresponding response variable Y
#(by subject ID)
library(dplyr)
Y1.MRI <- NULL
for (i in 1:nrow(nedd9_adni1_filter)) {
  Y1.MRI <-rbind(Y1.MRI, mri[mri$ID %in%
  row.names(nedd9_adni1_filter)[i],])
}

## corresponding Confounders (filter)
Z <- NULL
for (i in 1:nrow(nedd9_adni1_filter)) {
  Z <-rbind(Z, allchr[row.names(allchr) %in%
  row.names(nedd9_adni1_filter)[i],])
}

#Naive approach
Y <- scale(Y1.MRI[,-1]) #  q=56, n=64
X <- apply(nedd9_adni1_filter,2,function(y)(y -
mean(y))/sd(y)^as.logical(sd(y)))
```

```
Z <- as.matrix(scale(Z))

FF <- as.matrix(cbind(X, Z))

#multivariate linear regression
naive_model <- lm( Y ~ FF -1 )
GG <- naive_model$coefficients
C_ols <- GG[1:360, ] #360 feature coefficients
D_ols <- GG[361:370,] #10 PCs coefficients
D <- round(D_ols,2) #initialize D

## Try Cross-validation to choose lambda 1 and lambda 2

mygrid1 = c( 0.075, 0.1, 0.25, 0.5, 0.75) #tuning for B, l1
mygrid2 = c( 0.075, 0.1, 0.25, 0.5, 0.75) #tuning for o, l2

r <- 1
cvresults=matrix(NA,nrow=length(mygrid1),ncol=length(mygrid2))
for (i in 1:length(mygrid1))
  for(j in 1:length(mygrid2)) {
    cat(paste("lambda1",i,"  lambda2",j,"\n"))
    # In case the fitting in the CV fails,
    # wrap call to do_cv() in try()
    system.time({
    tem <- try(do_cv(c(mygrid1[i],mygrid2[j],r), Y, X, Z, D,
    num_folds=3))
    })
    cvresults[i,j]=ifelse(class(tem)=="try-error",NA,tem)
  }

rownames(cvresults) <- paste("lambda1=",mygrid1)
colnames(cvresults) <- paste("lambda2=",mygrid2)
cvresults
```

Appendix D

Appendix: Simulated Data

```
## Simulate dataset
#coded by Pulindu Ratnasekera, pratnase@sfu.ca
#Generate genotypes based on MAF calculated from 1000G data

setwd("/Users/gloriayang/Desktop/Cov-SRRR_Gloria")
source("aim.gen_1kG.R")

# Sub-population specific minor allele frequencies
# from 1000G data - chr 9 - 344575 markers
# CDX: Chinese Dai in Xishuangbanna, China
# CHB: Han Chinese in Bejing, China
# IBS: Iberian population in Spain
# FIN: Finnish in Finland

# rs4621895 - row 79789 of .frq files
CDX <- read.table(file='/Users/gloriayang/Desktop
/Cov-SRRR_Gloria/CDX_chr9.frq', header=T)
CDX_freq <- CDX$MAF
CHB <- read.table(file='/Users/gloriayang/Desktop
/Cov-SRRR_Gloria/CHB_chr9.frq', header=T)
CHB_freq <- CHB$MAF
FIN <- read.table(file='/Users/gloriayang/Desktop
/Cov-SRRR_Gloria/FIN_chr9.frq', header=T)
```

```
FIN_freq <- FIN$MAF
IBS <- read.table(file='/Users/gloriayang/Desktop
/Cov-SRRR_Gloria/IBS_chr9.frq', header=T)
IBS_freq <- IBS$MAF


# Retrieving Sub-population specific
#minor allele frequencies from 1000G data
#- chr 9 - 212313 markers (maf>0.05)
CDXind=CHBind=FINind=IBSind=rep(NA,length(CDX_freq))

for (i in 1:length(CDX_freq))
{
  if (CDX_freq[i]>0.05){CDXind[i]=1}else{CDXind[i]=0}
  if (CHB_freq[i]>0.05){CHBind[i]=1}else{CHBind[i]=0}
  if (FIN_freq[i]>0.05){FINind[i]=1}else{FINind[i]=0}
  if (IBS_freq[i]>0.05){IBSind[i]=1}else{IBSind[i]=0}
}

sum_ind  <- CDXind + CHBind + FINind + IBSind
ind      <- which(sum_ind==4)

CDX_freq <- CDX_freq[ind]
CHB_freq <- CHB_freq[ind]
FIN_freq <- FIN_freq[ind]
IBS_freq <- IBS_freq[ind]


# Simulated aims for given the number of makers
nm <- 10000 # number of markers
# nAIMs : number of markers to be simulated
# S_size: number of individuals in each sub-population
# freq  : minor allele frequency at each marker
CDX_aim  <- aim.gen(nAIMs=nm, S_size=100, freq=CDX_freq[1:nm])
CHB_aim  <- aim.gen(nAIMs=nm, S_size=100, freq=CHB_freq[1:nm])
```

```
IBS_aim   <- aim.gen(nAIMs=nm, S_size=100, freq=IBS_freq[1:nm])
FIN_aim   <- aim.gen(nAIMs=nm, S_size=100, freq=FIN_freq[1:nm])

aims <- rbind(CDX_aim, CHB_aim, IBS_aim, FIN_aim)
#dim(aims): 400 subj. x 10000 SNPs

## Confounders
library(factoextra)
pca_z <- prcomp(aims, center = T)
fviz_eig(pca_z)
Z <- pca_z$x[,1:10] #dim(Z):400 subj. x 10 PCs

## Predictors
set.seed(1)
X <- aims[,sample(1:10000, 1000, replace = F)]
#dim(X): 400 subj. x 1000 SNPs

## Simulated response
library(mvtnorm)
n <- 400; p <- 1000; q <- 4; r <- 1
#generate A,B,D,E to get Y
B<-round(c(rep(1,5),rep(0,p-5)),2) #B is p*r=p*1
A<-round(matrix(rep(0.4,q)),2) #A is q*r=q*1
C<- B %*% t(A)
D <- round(matrix(rep(1, ncol(Z)*q),
               nrow = ncol(Z),ncol = q),2)
SigE<-matrix(c(1,.9,0,0,.9,1,0,0,
               0,0,1,.9,0,0,.9,1), nrow=q,byrow=T)
E<-rmvnorm(n,sigma=.1*SigE)

Y <- X %*% B %*% t(A) +  Z %*% D + E
```